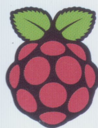
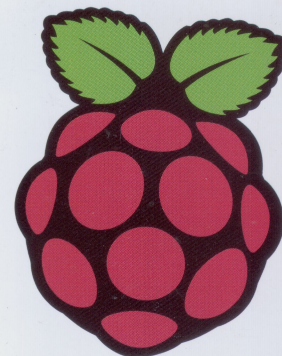
elektor

HET MAGAZINE GESCHREVEN DOOR EN VOOR DE RASPBERRY PI-COMMUNITY

Het officiële **RASPBERRY PI** magazine



MagPi



Nr. 08
mei-juni 2019

www.magpi.nl

Het officiële **Raspberry Pi**-magazine

ARCADE SPELLEN

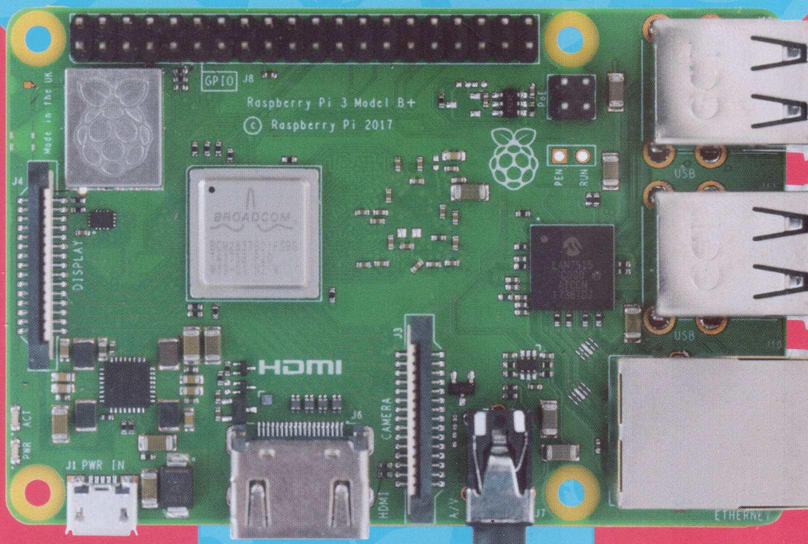




BOUW een flipperkast | **PROGRAMMEER** een 3d-spel | **SPEEL** pc-spelletjes

Plus!

- Bouw je eigen telefooncentrale **met Asterisk**
- 3d-modellen door tomografie **met de Raspberry Pi Camera Module**
- Een koelkast voor de overwintering van je schildpad **met temperatuursensor en zuurstofcontroller**



AI op je Raspberry Pi zonder internet

Automatisch objecten herkennen met de Raspberry Pi Camera Module en de Coral USB Accelerator van Google

Social API's

Integreer je software met Fortnite, weerstations en sociale netwerken

Nr. 08 mei-juni 2019 | € 9,95



WELKOM

Welkom bij het officiële Raspberry Pi Magazine

Sinds het begin al was de Raspberry Pi populair om er arcadegames op te spelen. Reden te meer om er deze keer een extra lang dossier aan te wijden. Vanaf **pagina 14** leer je hoe je een flipperkast bouwt, een 3d-spel programmeert en pc-games speelt. En het bouwen en programmeren is zeker zo leuk als het spelen!

Verder besteden we in dit nummer ook in enkele artikels aandacht aan 'lokale AI': die draait op je computer zelf in plaats van in de cloud. Het voordeel? Je dient geen privacygevoelige informatie prijs te geven en je hebt geen last van de vertraging van het netwerk.

Op **pagina 12** lees je mijn interview met Joseph Dureau, de CTO van Snips. Het bedrijf heeft een stemassistent ontwikkeld die je privacy respecteert en op een Raspberry Pi draait. Bij mij draait Snips al een tijdje in de woonkamer!

Verrassend genoeg ziet ook Google, dat onder andere met zijn Google Assistant een cloudgebaseerde stemassistent aanbiedt, heil in deze aanpak. Op **pagina 8** bespreken we de Coral USB Accelerator, een apparaatje dat je op je Raspberry Pi aansluit om neurale netwerken te versnellen, en op **pagina 66** tonen we je hoe je je Raspberry Pi ermee voorwerpen laat herkennen.

Uiteraard hebben we zoals elke keer ook weer vele praktijkartikelen en projecten voor je klaarstaan. Zodat je kunt bouwen, programmeren en spelen.

Koen Vervloesem Hoofdredacteur MagPi



HOOFDREDACTEUR

Koen Vervloesem

Hoofdredacteur van MagPi. Houdt van programmeren en zijn huis vol elektronica steken. Zijn schaarse vrije tijd besteedt hij aan sporten, lezen, tuinieren en in de natuur wandelen.

magpi.nl

VIND ONS ONLINE magpi.nl

NEEM CONTACT OP redactie@magpi.nl

MagPi

REDACTIE
Hoofdredactie: Koen Vervloesem (koen@magpi.nl)
Redactie: Lucy Hattersley, Rob Zwetsloot, Phil King en Jem Roberts
Medewerkers: Alex Bate, Mike Cook, David Crookes, Sean McManus, Laura Sachs, Richard Smedley, Clive Webster

ADVERTENTIES
Margriet Debeij
magriet.debeij@elektor.com

ABONNEMENTEN
abonnementen@magpi.nl
www.elektor.nl/magpi-abo

UITGEVER
Elektor International Media B.V.
Postbus 11, 6114 ZG Susteren, Nederland
Tel.: +31 (0)46- 4389444
Don Akkermans (Directeur)



VORMGEVING
Dougal Matthews, Julian van den Berg



DRUK: Senefelder, Doetinchem
DISTRIBUTIE: Betapress, Gilze



MagPi, het officiële Raspberry Pi Magazine (Nederlandstalige editie), is een uitgave onder licentie van MagPi, gepubliceerd door Raspberry Pi (Trading) Ltd, 30 Station Road, Cambridge, CB12JH, Verenigd Koninkrijk.

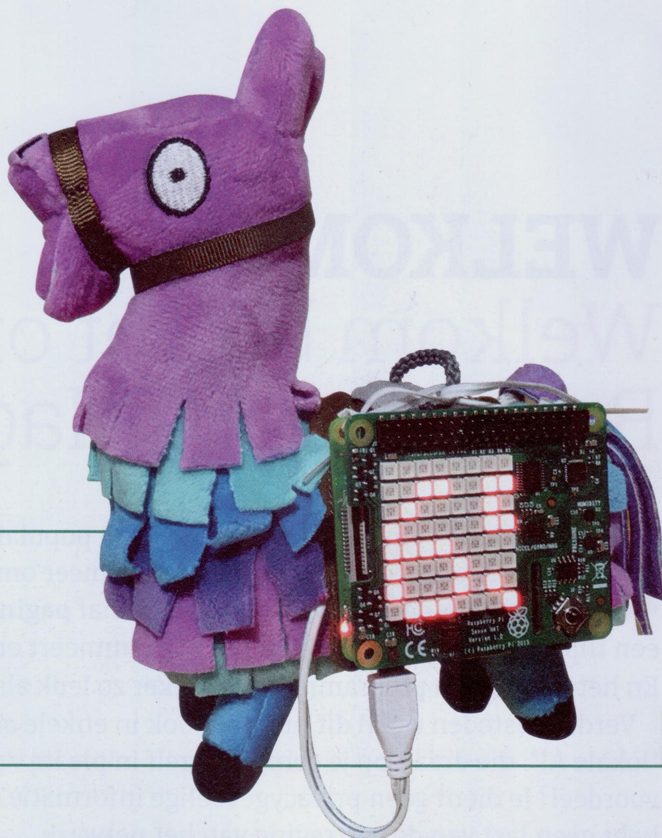
 ISSN 2589-5214.



Inhoud

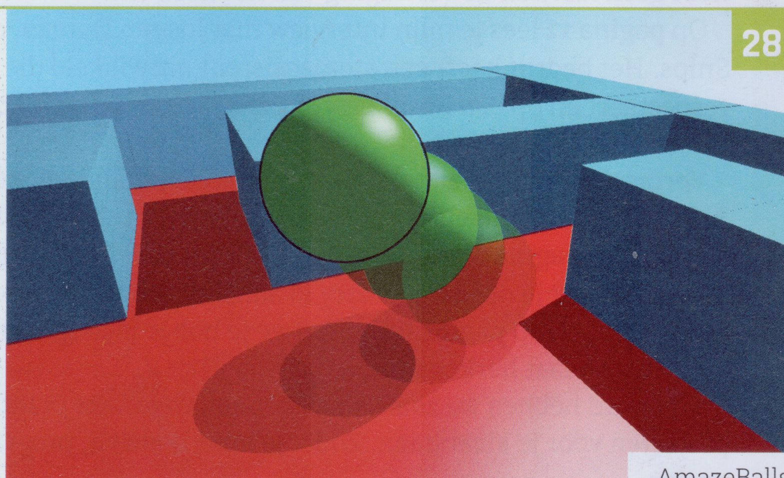
> Nr. 08 > mei-juni 2019

Hoofdartikel



84

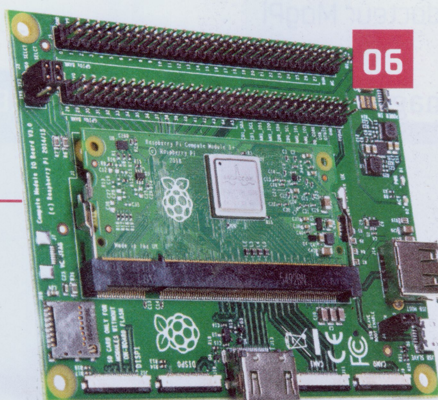
Praktijk



28

AmazeBalls

Nieuws



06

De Compute Module 3+



56

Bouw je eigen telefooncentrale





Raspberry Pitaar





Inky wHAT

Dossier

Projecten

Tests

Community

Diversen

De Compute Module 3+

De Raspberry Pi Foundation onthult zijn nieuwste industriële computer. Door **Lucy Hattersley**.

De Raspberry Pi Foundation heeft zijn Compute Module 3+ uitgebracht met technische verbeteringen geïnspireerd door andere recentere Pi-bordjes.

De Compute Module richt zich op bedrijven en industriële gebruikers en dient om een Raspberry Pi in industriële systemen en commerciële producten op te nemen. Het bordje heeft het formaat van een standaard DDR2 SODIMM (*small outline dual in-line memory module*). De gpio-pennen en andere I/O-functies zijn beschikbaar via de 200 pennen op het bord.

De nieuwe Compute Module 3+ is uitgerust met de Broadcom BCM2837B0, zij het op een lagere kloksnelheid van 1,2 GHz. "We hebben de kloksnelheid niet verhoogd, omdat dit afhangt van de MaxLinear PMIC, die niet in het formaat van de Compute Module 3+ past", legt Eben Upton uit, ceo en mede-oprichter van de Raspberry Pi Foundation. "Zoals altijd proberen we lering te trekken uit de markt." De 1 Gbyte LPDDR2 SDRAM is wel hetzelfde als in zijn niet-industriële broertje, de Raspberry Pi 3B+.

Het doel van deze update van de Compute Module is niet om sneller te werken, maar koeler: "De Compute Module 3+ brengt ons verbeterde thermische prestaties en wat kleine aanpassingen aan de printplaat die van de Raspberry Pi 3B+ komen."

Specificaties van de Compute Module 3+

Processor:
Broadcom BCM2837B0, Cortex-A53 64-bits SoC 1,2 GHz

Geheugen:
1 Gbyte LPDDR2 SDRAM

Flashgeheugen (eMMC):
8/ 16 / 32 Gbyte

Multimedia:
H.264, MPEG-4 decode (1080p30), H.264 encode (1080p30), OpenGL ES 1.1 en 2.0

Bedrijfstemperatuur:
-20 to +70°C

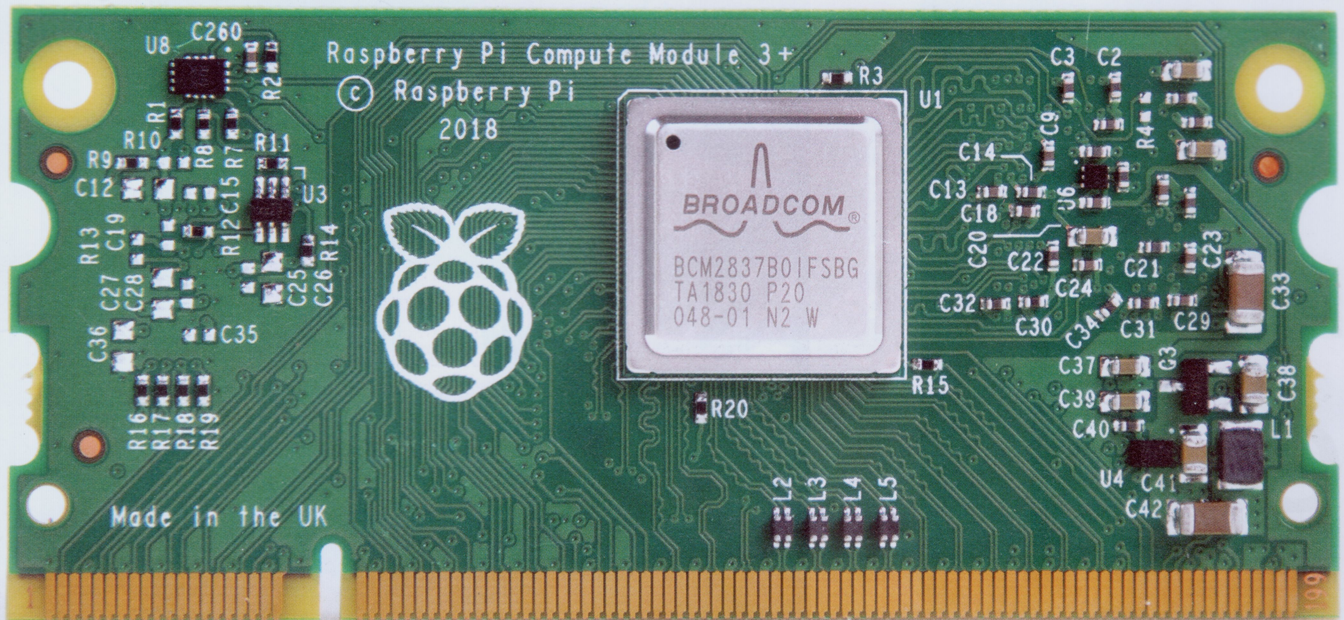
Prijs:
€ 49,95 voor het 32 Gbyte-model

Nettogewicht:
9 g (alleen het bord)

Afmetingen:
67,6 × 31,1 × 3,7 mm (alleen het bord)



▼ Het Compute Module I/O-bord helpt ontwikkelaars door alle poorten te voorzien die je op een Raspberry Pi vindt.



▲ De Broadcom BCM2837B0 en de aanpassingen aan de printplaat laten het bordje toe om in een breder temperatuurbereik te werken.

Een koel hoofd

De Compute Module 3+ is een "kans om de Raspberry Pi in omgevingen met een breder temperatuurbereik in te zetten", zegt Eben. "Ingenieurs kunnen de chip zwaarder en langduriger belasten, zonder dat ze tegen de thermische limieten aanlopen." Het bereik voor de bedrijfstemperatuur gaat nu van -20 tot +70 graden Celsius.

> "Belast de chip zwaarder en langduriger zonder tegen de thermische limieten aan te lopen"

De opslag op het bordje is in het nieuwe model ook uitgebreid. Er zijn nu versies met 8, 16 en 32 Gbyte eMMC-flashgeheugen.

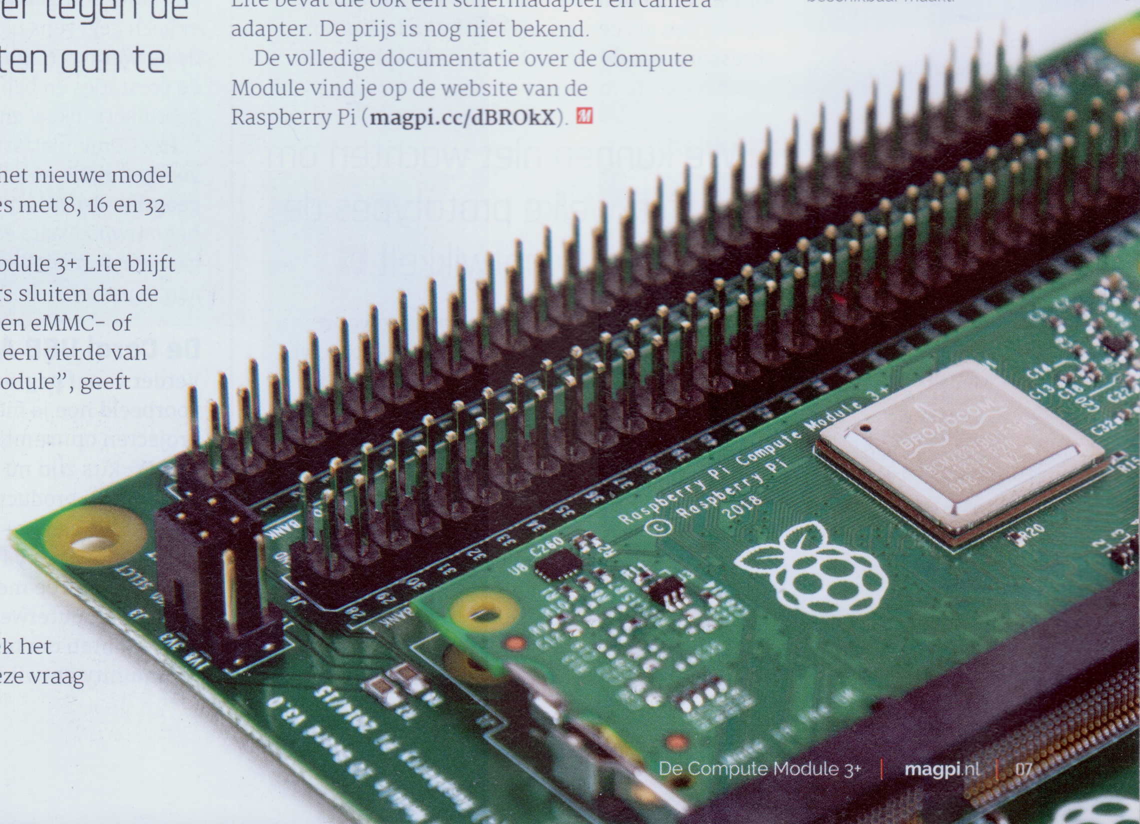
De optie van een Compute Module 3+ Lite blijft ook nog beschikbaar. Gebruikers sluiten dan de pennen van de module aan op een eMMC- of SD-kaart. "De Lite is goed voor een vierde van de verkopen van de Compute Module", geeft Eben prijs.

Klanten vragen al even naar opties met meer opslag, en de nieuwe modellen zullen dan ook zeker populair zijn bij ontwikkelaars van apparaten voor multimedia en datalogging. "Nu de prijzen van flash aan het dalen zijn, leek het ons een goed moment om op deze vraag in te gaan", zegt Eben. De versie met 32 Gbyte flashgeheugen is bij Elektor te koop voor € 49,95 (**www.elektor.nl/rpi-cm3-32gb**).

Je kunt een afzonderlijk Raspberry Pi Compute Module I/O-bord kopen om te helpen bij de ontwikkeling. Dat break-outboard levert de poorten die je op een normale Raspberry Pi zou vinden, zoals gpio-pennen, hdmi, usb, camera en scherm.

Er komt ook een Raspberry Pi CM3+ Development Kit. Naast een Raspberry Pi Compute Module I/O-bord, een CM3+/32 Gbyte en een CM3+/Lite bevat die ook een schermadapter en camera-adapter. De prijs is nog niet bekend.

De volledige documentatie over de Compute Module vind je op de website van de Raspberry Pi (**magpi.cc/dBROkX**).



▼ De Compute Module komt in een standaard DDR2 SODIMM-formaat dat de gpio-pennen en andere I/O-functies via de 200 SODIMM-pennen beschikbaar maakt.

coral.withgoogle.com

Google brengt Coral uit

Een nieuw platform om apparaten te bouwen met lokale AI.

Google heeft met Coral een nieuw platform hardwarecomponenten en softwarehulpmiddelen gelanceerd met de Edge TPU. Daarmee kun je neurale netwerken lokaal versnellen. De Coral USB Accelerator is ontworpen als een gemakkelijk aan te sluiten accessoire om de Edge TPU op de Raspberry Pi beschikbaar te maken.

> We kunnen niet wachten om te zien welke prototypes de community ontwikkelt

"We zijn verheugd dat we nu een nieuwe productlijn kunnen introduceren die voortbouwt op ons werk met de originele AIY-kits. Het zijn producten waarmee je neurale netwerken in je eigen projecten kunt opnemen met aandacht voor privacy, snelheid en energiezuinigheid", zegt Billy Rutledge, directeur van de AIY- en Coral-teams bij Google.

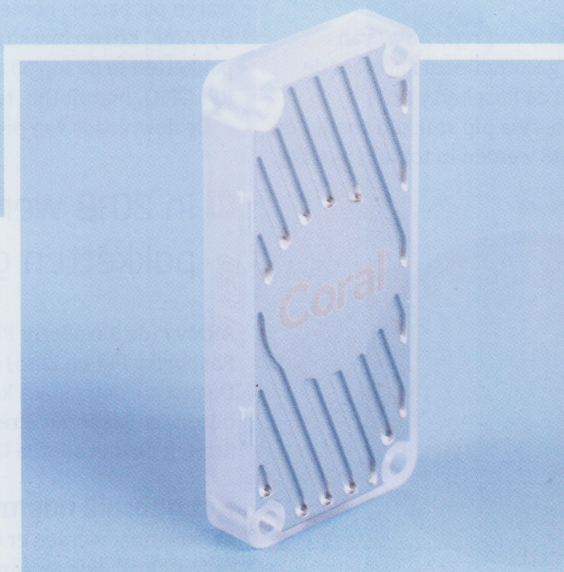
Deze neurale netwerken die op het apparaat zelf draaien, laten makers toe om snelle en efficiënte AI-mogelijkheden aan hun prototypes en projecten toe te voegen met behoud van hun privacy. Ontwikkelaars creëren en trainen hun neurale netwerken met TensorFlow voordat ze die compileren en op de Edge TPU-bordjes draaien met de meegeleverde software. Zodra het gecompileerde netwerk geïnstalleerd is, gebeuren alle berekeningen lokaal op de Edge TPU en worden er geen gegevens naar de cloud gestuurd. Zonder de netwerkvertraging van de cloud verhogen de prestaties en behoud je de gegevens van de gebruikers lokaal en onder je eigen controle.

Het Coral-platform bevat ook een verzameling voorgetrainde en voorgecompileerde modellen die geoptimaliseerd zijn voor de Edge TPU. Ze maken het ontwikkelaars eenvoudig om aan de slag te gaan en de modellen zijn ook voor eigen gebruik aan te passen.

De Coral USB Accelerator gebruiken

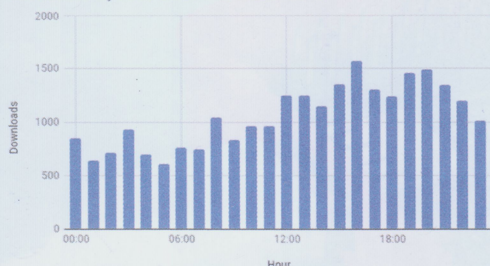
Verder vanaf **pagina 66** tonen we je in een voorbeeld hoe je met Coral lokale AI in je eigen projecten opneemt. De nieuwe Coral-producten en AIY-kits zijn nu wereldwijd te koop.

De Coral-productlijn is een welkome toevoeging aan het ontwikkelgereedschap voor de Raspberry Pi. Artificial Intelligence blijft een van de meest creatieve domeinen van de computerwereld. We kunnen dan ook niet wachten om te zien welke prototypes de community ontwikkelt.



▲ Verbind de Coral USB Accelerator met je Raspberry Pi via een kabel die USB type A naar USB type C omzet.

◀ De Coral USB Accelerator van Google bevat een Edge TPU-chip.







Veel tijd bespaard met piwheels

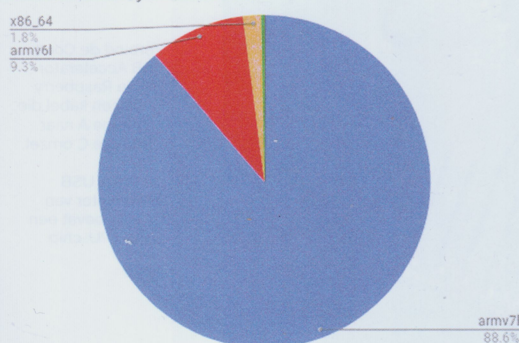
Het project piwheels heeft gebruikers van de Raspberry Pi heel wat gemakkelijker gemaakt om Python-projecten te downloaden.

Het project piwheels heeft gegevens over de downloads van Python-pakketten in 2018 en hoeveel tijd het project gebruikers heeft bespaard. Door **Ben Nuttall**.

Het project piwheels heeft het gebruikers van de Raspberry Pi heel wat gemakkelijker gemaakt om Python-projecten te downloaden.

Het project piwheels is een repository van Python-projecten die gecompileerd zijn voor de ARM-architectuur van de Raspberry Pi.

Het maakt installaties via pip stukken sneller op de Raspberry Pi. In 2018 werden in totaal 5.154.233 pakketten gedownload, wat alle gebruikers samen 39 jaar, 10 maanden en 29 dagen heeft uitgespaard!



De top 30 downloads

De vijf meest gedownloade pakketten van 2018 waren pycparser, home-assistant-frontend, PyYAML, cffi en MarkupSafe. Andere belangrijke pakketten in de top 30 zijn numpy, cryptography, RPi.GPIO, matplotlib, tensorflow en opencv.

De downloads van piwheels waren 89% voor ARMv7 (de Raspberry Pi 2 en 3), 9% voor ARMv6 (de Raspberry Pi 1 en Zero) en 2% andere architecturen. De meeste downloads komen van Raspbian, in het bijzonder Raspbian Stretch. Daarna komen Debian Stretch en dan allerlei Ubuntu-distributies.

> „In 2018 werden in totaal 5.154.233 pakketten gedownload"

Het gebruik neemt toe

In december werden er 662.621 pakketten gedownload, wat op één maand al meer dan zes jaar tijd heeft uitgespaard.



▲ Het officiële Raspberry Pi-toetsenbord is een volledig toetsenbord met 79 toetsen en functioneert ook als usb-hub.

Officieel toetsenbord en muis

De Raspberry Pi heeft nu een officieel toetsenbord en muis. Door **Lucy Hattersley**.

De Raspberry Pi Foundation heeft twee nieuwe officiële apparaten uitgebracht voor de Raspberry Pi: enerzijds een toetsenbord dat ook als hub dient en anderzijds een muis. Beide zijn speciaal voor gebruik met de Raspberry Pi ontworpen.

Het officiële Raspberry Pi-toetsenbord is een volledig toetsenbord met 79 toetsen dat je via een USB-type-A-naar-type-B-kabel aansluit.

Op de achterkant van het toetsenbord vind je nog eens drie USB-type-A-poorten, die van het toetsenbord dus ook een hub maken. Ideaal om extra usb-apparaten aan te sluiten, zoals de nieuwe muis.

Rood en wit

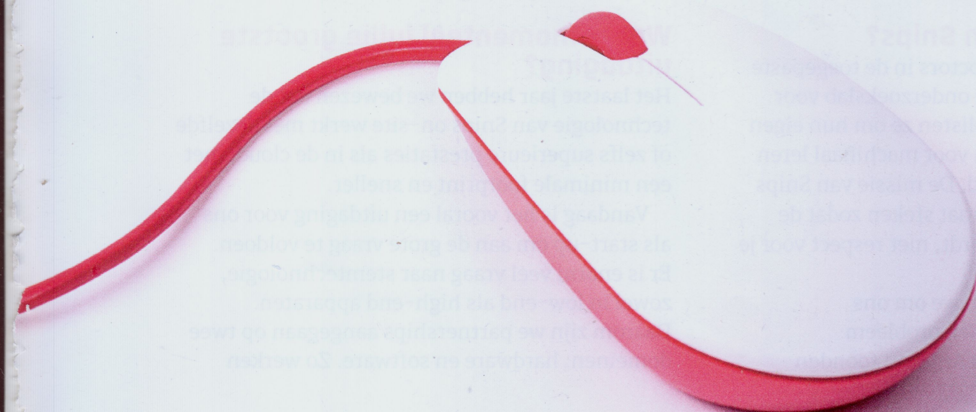
Het hoogkwalitatieve toetsenbord is ergonomisch en past netjes op je tafelblad. Het is wit met rode accenten, wat past bij de officiële behuizing en andere producten voor de Raspberry Pi.

De officiële muis — eveneens in rood en wit — is een optische muis met drie knoppen en een scrollwieltje. Je verbindt het eveneens met je Raspberry Pi via een USB-type-A-connector. Beide apparaten verbind je met een draad in plaats van Bluetooth. Dat maakt het eenvoudiger om ze te configureren en je dient je ook geen zorgen te maken over batterijen.

Beide apparaten zijn binnenkort te koop.

> Beide apparaten verbind je met een draad

▼ De nieuwe officiële muis ziet er geweldig uit naast het toetsenbord en een Raspberry Pi.



Interview met Joseph Dureau, CTO van Snips

De Parijse start-up Snips wil stemtechnologie in elk huishouden brengen. Zijn stemassistent die je privacy respecteert draait volledig lokaal op een Raspberry Pi. **Koen Vervloesem** sprak met CTO Joseph Dureau.



▲ Joseph Dureau, CTO van Snips

Vorig jaar schreven we in MagPi voor het eerst een kort nieuwsbericht over Snips (**snips.ai**). Al die tijd bleef de offline stemassistent op onze radar staan. Tijd om weer eens te kijken hoe ver het met dit interessante stukje AI-technologie staat, vonden we. Daarom interviewden we Joseph Dureau, CTO van Snips, over de missie van het bedrijf, over wat Snips onderscheidt van Google en Amazon, en over het belang van de Raspberry Pi voor Snips.

Wat is de missie van Snips?

Snips is in 2013 door drie doctors in de toegepaste wiskunde opgericht als een onderzoekslab voor grote bedrijven. Al snel beslisten ze om hun eigen technologie te ontwikkelen voor machinaal leren zonder gebruik van de cloud. De missie van Snips is sindsdien: AI in elk apparaat steken zodat de technologie onzichtbaar wordt, met respect voor je privacy.

Drie jaar geleden besloten we om ons toe te spitsen op een specifiek probleem: stemassistenten. Google en Amazon toonden de mogelijkheden van stemassistenten als nieuwe interface, maar dat bracht allerlei privacy-implicaties met zich mee. Zo wordt elke spraakopdracht van je naar de cloud verzonden. Bovendien dringen deze bedrijven zichzelf op als interface tussen de gebruikers en de systemen erachter: gebruikers spreken met "Google" of "Alexa", niet met het merk van het product dat de stemassistent integreert.

Dat is waar wij met onze stemassistent het verschil maken: Snips implementeert niet alleen *privacy by design* door je stem lokaal te verwerken, maar is ook een white-labelproduct: gebruikers spreken niet met "Snips", maar met het merk van het product dat Snips integreert. We hebben momenteel contracten gesloten met meer dan 40 bedrijven die onze stemassistent in hun producten gaan inbouwen. In 2019 komen er domoticatoestellen, apparaten voor slimme gebouwen, financiële diensten en apparaten voor "industrie 4.0" op de markt met onze technologie in.

> " De missie van Snips: AI in elk apparaat steken zodat de technologie alomtegenwoordig wordt, met respect voor je privacy "

Wat is momenteel jullie grootste uitdaging?

Het laatste jaar hebben we bewezen dat de technologie van Snips on-site werkt met dezelfde of zelfs superieure prestaties als in de cloud, met een minimale footprint en sneller.

Vandaag is het vooral een uitdaging voor ons als start-up om aan de grote vraag te voldoen. Er is enorm veel vraag naar stemtechnologie, zowel in low-end als high-end apparaten. Daarom zijn we partnerships aangegaan op twee domeinen: hardware en software. Zo werken

we op hardwaregebied onder andere samen met chipproducent NXP aan een referentieontwerp voor stemtechnologie. En we werken ook samen met diverse integratiepartners die onze technologie gebruiken om voor onze klanten een steminterface van superieure kwaliteit te ontwikkelen.

Hoe belangrijk is de Raspberry Pi in jullie inspanningen?

De Raspberry Pi heeft ons enorm geholpen om onze oplossing aan iedereen aan te bieden. Snips draait op de Raspberry Pi en is gratis voor niet-commercieel gebruik. In samenwerking met Seeed bieden we ook Pi-gebaseerde Voice Interaction ontwikkelkits aan. Iedereen kan daardoor eenvoudig met onze technologie aan de slag.

Het was ons vanaf het begin duidelijk dat de Pi de juiste oplossing is om Snips in ieders handbereik te krijgen. Bovendien bevat de Pi 3 exact het type hardware dat je in de IoT-wereld vindt, in een tv of set-topbox. Op die hardware richten we ons.

Hebben al veel mensen met Snips geëxperimenteerd?

Momenteel hebben al 25.000 ontwikkelaars zich geregistreerd op onze webgebaseerde Snips Console, en dat zonder enige marketing. Het is geweldig om te zien wat ze allemaal doen met ons platform, zoals alle apparaten in hun huis met hun stem aansturen. Ze sturen ons video's over hun experimenten en ze zijn heel veeleisend als er iets niet werkt.

Dat zoveel mensen ons platform waardevol vinden, is ook omdat we open zijn over onze ontwikkelingen. We publiceren wetenschappelijke artikels, we beschrijven hoe ons platform werkt en we hebben een deel van onze software (Snips NLU) opensource gemaakt. De broncode van andere delen volgt nog.

Wanneer kunnen we in het Nederlands met Snips spreken?

Dat zal in 2019 niet meer gebeuren, vrees ik, maar zeker in 2020. We willen tegen volgend jaar 50% van de talen in de wereld ondersteunen, en zeker de belangrijkste Europese talen. Momenteel herkent onze assistent Frans, Engels, Duits, Spaans, Italiaans en Japans.

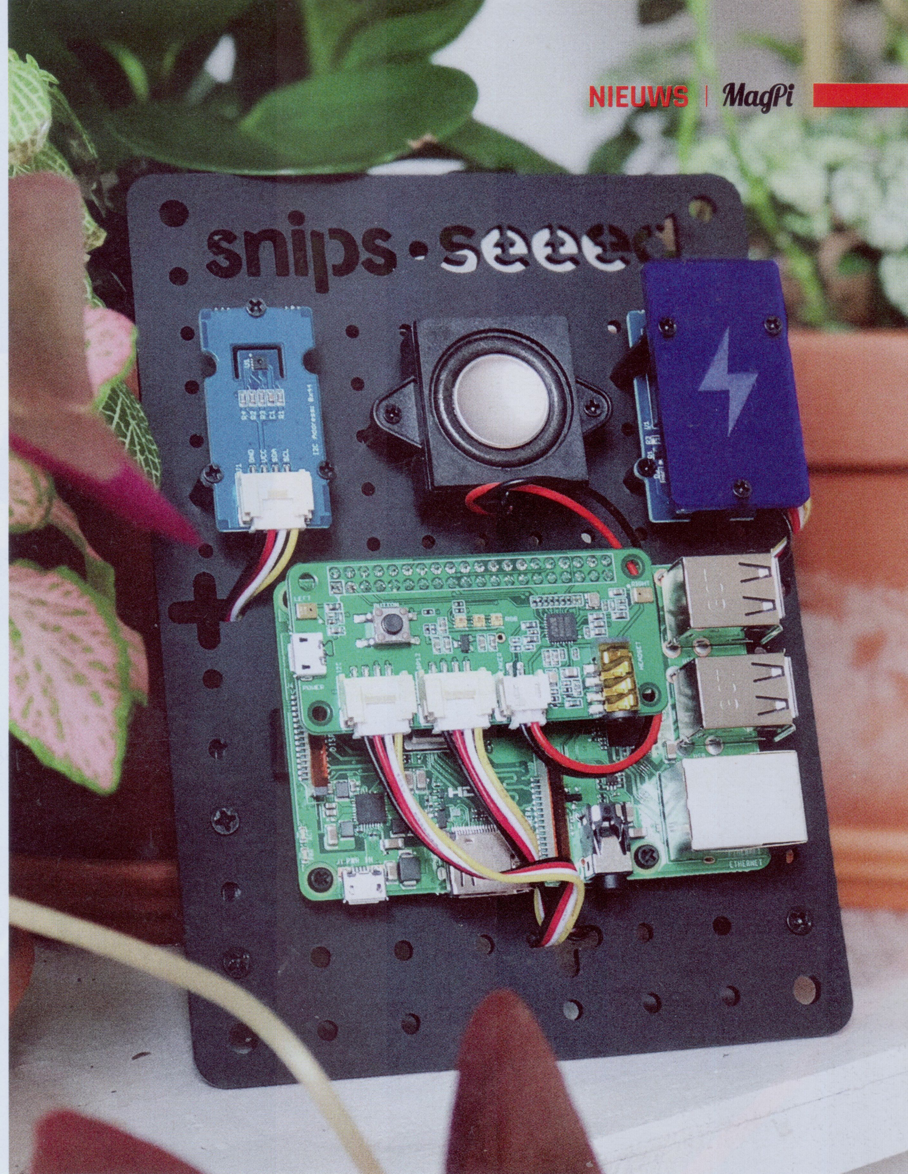
Een nieuwe taal ondersteunen is 3 tot 6 maanden werk. Een expert in die taal dient de grammatica te schrijven, een akoestisch model te trainen enzovoort. Het is best ingewikkeld.

En wat kunnen we in de toekomst nog van Snips verwachten?

Momenteel focussen we ons op twee domeinen. Allereerst willen we de mogelijkheden op apparaten zoals de Raspberry Pi 3 uitbreiden. Een grote woordenschat is al mogelijk, maar we willen nu ook de prestaties van meerdere woordenschatdomeinen op dezelfde assistent verbeteren, zoals recepten, routebeschrijvingen, algemene kennis, ...

Daarnaast kijken we ook maar microcontrollers. We brachten onlangs Snips Commands uit, dat op een ARM Cortex-M4 en Cortex-M7 draait. Dit soort apparaten heeft niet de verwerkingskracht voor een grote woordenschat of meerdere domeinen, maar ze zijn wel goed om eenvoudige opdrachten zoals "speel", "pauzeer", "stop" enzovoort te begrijpen. We blijven proberen om de grenzen van deze kleine apparaten te verleggen.

Op langere termijn willen we overal in huis apparaten plaatsen die met elkaar communiceren, zodat het heel normaal wordt om met al je apparaten te spreken. Dit noemen we Snips AIRnet. We willen de stemassistent ook een geheugen geven, zodat er veel natuurlijkere en relevantere interacties mogelijk zijn. In een cloudgebaseerde stemassistent ga je geen geheugen inbouwen: dat zou een echte privacynachtmerrie zijn.



▲ De Voice Interaction ontwikkelkits die Snips in samenwerking met Seeed ontwikkelde, laten toe om eenvoudig met een stemassistent te experimenteren.

ARCADE SPELLEN

BOUW een flipperkast | **PROGRAMMEER** een 3d-spel | **SPEEL** pc-spelletjes

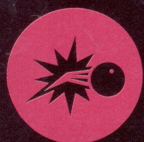


Probeer deze fantastische spelprojecten uit op je Raspberry Pi

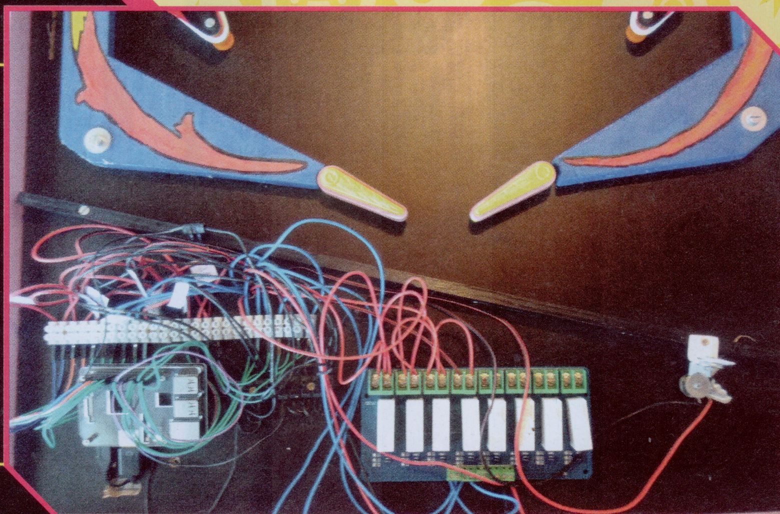
Er bestaan talloze manieren om met je Raspberry Pi computerspelletjes te spelen. Of je nu een fysiek spel bouwt, je eigen spel programmeert of eenvoudigweg pc-games op je Raspberry Pi wilt spelen, je favoriete minicomputer is een geweldig stukje hardware voor zowel videospelletjes als fysieke spelletjes.

We hebben in dit dossier drie projecten bij elkaar gebracht die je een voorproefje van dit alles geven. Eerst leggen we je uit hoe je een flipperkast bouwt uit een oud kinderbed. Daarna tonen we je hoe je Steam-games op je Raspberry Pi speelt. En tot slot programmeer je je eigen avonturenspel in 3d met Pygame Zero.

Klaar om te spelen?

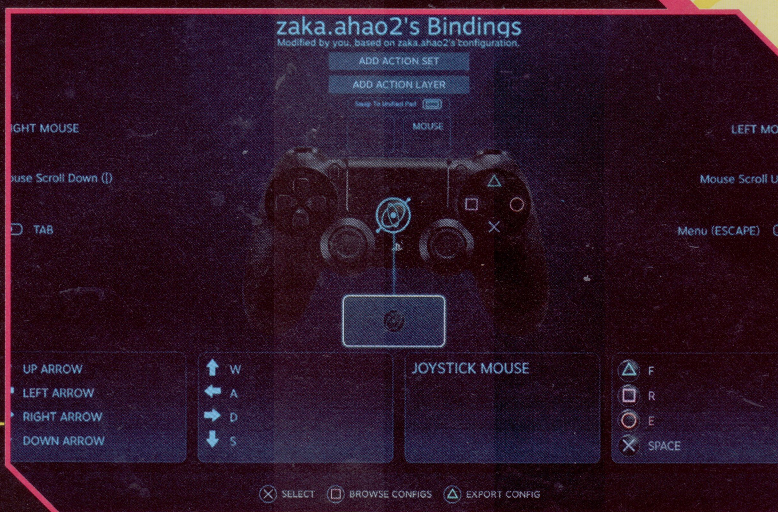


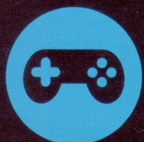
BOUW EEN FLIPPERKAST
PAGINA 16



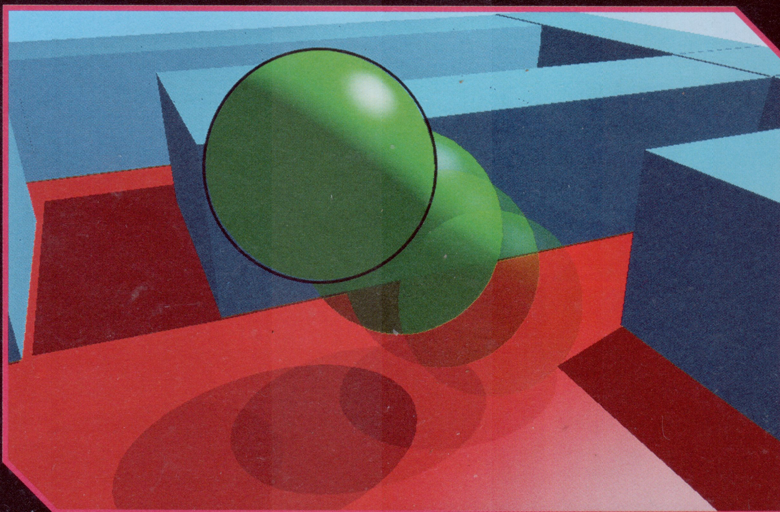


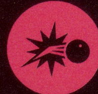
SPEEL STEAM-SPELLETJES
PAGINA 24

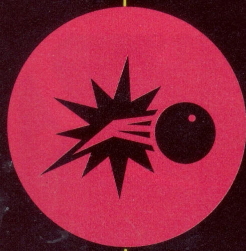




PROGRAMMEER JE EIGEN 3D-SPEL
PAGINA 28







FLIPPER KONING

Toen de dochter van **Martin Kauss** uit haar prinsessenbedje groeide, transformeerde hij het tot een geavanceerde flipperkast. In dit artikel leer je hoe je hetzelfde ook kunt.

Opgroeiende kinderen laten een spoor van afval achter in hun kielzog, maar er is een speciaal soort maker voor nodig om een oud decoratief bedframe te zien en zich dat in een speelhal in te beelden.

We spraken met Martin Kauss over hoe hij van het bed van zijn dochter een unieke flipperkast maakte. Daarna tonen we in een stap-voor-stap gids hoe je zelf zo'n flipperkast in elkaar tovert.

Wat bezielde je om dit te doen?
Ik zocht een educatief project voor mezelf om interactie met de echte wereld via de gpio's te leren. Bovendien wilde ik iets inspirerends voor mijn kinderen maken dat leuk zou zijn om te bouwen en om mee te spelen.
Omdat ik een software-ontwikkelaar met een achtergrond in werktuigkunde ben, wilde ik iets echts bouwen. En omdat ik niet veel ervaring met elektronica had, was dit project tegelijk een introductie daarin.





K.G. Orphanides

K.G. is een schrijver, ontwikkelaar en maker die nog altijd vol ontzag is over de geweldige projecten die uit de Raspberry Pi-community komen.

Hoe lang heb je eraan gezeten?
In totaal moet het een drietal maanden zijn. Dat is inclusief voorbereiding, bestellingen, leren, bouwen, ontwikkelen, testen en optimaliseren.

Hoeveel plande je vooraf?
Om eerlijk te zijn: niet veel. Ik voerde alles in kleine stappen uit. Voor elke stap deed ik wel vooraf heel wat onderzoek, maakte ik prototypes en testte ik ze.

Zodra ik één onderdeel begreep (bijvoorbeeld het mechanisme van de flippers), testte ik de mechanische onderdelen tot ze werkten, waarna ik overging naar de software-ontwikkeling in combinatie met de nodige bevestiging en bedrading. Uiteindelijk deed ik dan de functionele tests.

Focussen op één component tot die werkte en dan naar de volgende component gaan werkte heel goed voor mij.

Wat was het moeilijkste deel van dit project?
Ik had geen enkele ervaring met flipperkasten, behalve dan dat ik in mijn kindertijd op enkele gespeeld heb. Het moeilijkste deel was voor mij dan ook de speciale woordenschat vinden die leidt tot de bronnen die me componenten zoals het flippermechanisme helpen begrijpen.

Het flippermechanisme was niet eenvoudig, en ook nog eens gevaarlijk — zelfs met mijn (in vergelijking met echte flipperkasten) laagvermogenvoeding van 36 V. Ik brandde een weerstand door in het proces. En met doorbranden bedoel ik echt vlammen... ook al waren het maar heel kleine die niet lang duurden.

Wat staat er nog op de planning?
Ook al heeft de Raspberry Pi vele gpio-pennen, ik heb er nog maar een paar over. Ik kan nog maar één jetbumper toevoegen of wat doelen. De beslissing is nog niet gevallen. Maar het speelveld heeft nog wat meer componenten nodig, zoals rubberen ringen.

Aantal	Component	Prijs per stuk	Prijs
3	Flipperbal	€ 3,12	€ 9,36
1	Kit met rubberen ringen	€ 31,43	€ 31,43
1	Shooter	€ 31,43	€ 31,43
2	Moer voor flipperknop	€ 1,89	€ 3,78
2	Flipperknop	€ 5,26	€ 10,51
1	Deksel slingshot (links	€ 10,41	€ 10,41
1	Deksel slingshot (rechts)	€ 10,41	€ 10,41
2	Flipperknop	€ 14,50	€ 29,00
3	Roll-over microschakelaars	€ 4,41	€ 13,23
1	Microschakelaar	€ 4,41	€ 4,41
1	Montage voor de spinner	€ 29,39	€ 29,39
1	Flippermontage met spoel FL-11630, normaal open EOS-schakelaar (links/rechts)	€ 51,48	€ 51,48
15	Starposts	€ 1,04	€ 15,61
2	Popbumpers	€ 43,29	€ 86,59
1	100 stuks krimpkousen	€ 4,98	€ 4,98
1	5 m 0,75 mm² elektrische draad in meerdere kleuren	€ 12,92	€ 12,92
1	36 V-voeding (TDK-Lambda LS150-36)	€ 45,85	€ 45,85
1	40 × 20 cm mannelijk/vrouwelijk jumperdraden	€ 5,52	€ 5,52
1	2 m ledstrip	€ 9,44	€ 9,44
1	5 V-voeding (TOOGOO AC 110 V/220 V)	€ 13,50	€ 13,50
1	SainSmart 8-kanaals 5 V halfgeleiderrelais	€ 14,74	€ 14,74
1	SainSmart 2-kanaals DC/DC 5 V/220 V 5 A halfgeleiderrelais	€ 12,89	€ 12,89

TOTAL: € 456,88

Omdat het in dit project allemaal gaat om het creëren van een unieke flipperkast, zal jouw onderdelenlijst niet identiek zijn aan die van Martin Kauss. Maar deze lijst helpt je wel om uit te zoeken welke componenten je voor je eigen project nodig hebt, en hoeveel ze ongeveer zullen kosten.

De gespecialiseerde componenten voor de flipperkast komen rond de € 450 uit. Maar dan tellen we nog niet het gereedschap, de onderdelen om de kast zelf te construeren en wat extra Pi- en gpio-componenten mee.

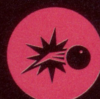
Voor de kast zelf heb je een frame nodig — in dit voorbeeld het prinsessenbed van Martins dochter — en een multiplexplank. Ook houten en aluminium sierlijsten van allerlei afmetingen voor een deur — daarmee maak je onderdelen zoals de schietbaan, smalle doorgangen en de poten. En verder nog stelschroeven met plastic voetjes voor de voorpoten van de flipperkast en een heleboel schroeven, inslagmoeren, bouten en montagebeugels om alles bij elkaar te houden.

Voor de elektronische en mechanische componenten zegt Martin dat zijn extra voorraad bestaat uit ledlampen, kroonstenen, krimpkousen, gpio-kabels, een extra grote gpio-header voor de Pi en (trek)veren (bijvoorbeeld om de schietbaan te sluiten).

Tot slot heb je ook een redelijk goed uitgeruste gereedschapskist nodig, inclusief soldeerbout, soldeerdraad, een multimeter, kabelstripper, schroevendraaiers, zaag, boor en bits, een hamer, een vijl en een 'helpende hand' met krokodilklemmen om dingen op hun plaats te houden terwijl je elektronische componenten in elkaar steekt.

Nuttige winkels

- flipperwinkel.nl
- pinballshop.nl
- ministryofpinball.com
- flipperteile.de
- flipperservice.de
- pinballlife.com
- sainsmart.com
- pinball.co.uk
- pinparts.co.uk



MAAK JE EIGEN FLIPPERKAST

Deze stap-voor-stap gids leidt je door de belangrijkste fases van de bouw van de prinsessenflipperkast. Hoewel jouw flipperkast heel verschillend kan zijn, blijven de belangrijkste componenten en technieken hetzelfde.

Elke flipperkast heeft een shooter (waarmee je de bal in het spel schiet), flippers, bumpers en rubbers nodig. En onze bouwtips — zoals verstelbare poten gebruiken om je te helpen de perfecte hoek op je flipperkast te krijgen — kun je op heel diverse bouwstijlen toepassen.

Ook de configuratie die Martin Kauss voor zijn gpio-verbindingen koos en de software die hij ontwikkelde om de lichten, sensoren, geluiden en scores aan te sturen, zijn krachtige hulpmiddelen voor je flipperkast.

01 Installeer de software

Begin met een propere installatie van Raspbian op een Pi. Je hebt een internetverbinding nodig. Een beeldscherm, toetsenbord en muis zijn niet strikt noodzakelijk, maar maken het je leven wel makkelijker. Op het beeldscherm kunnen we later ook de score van de speler weergeven.

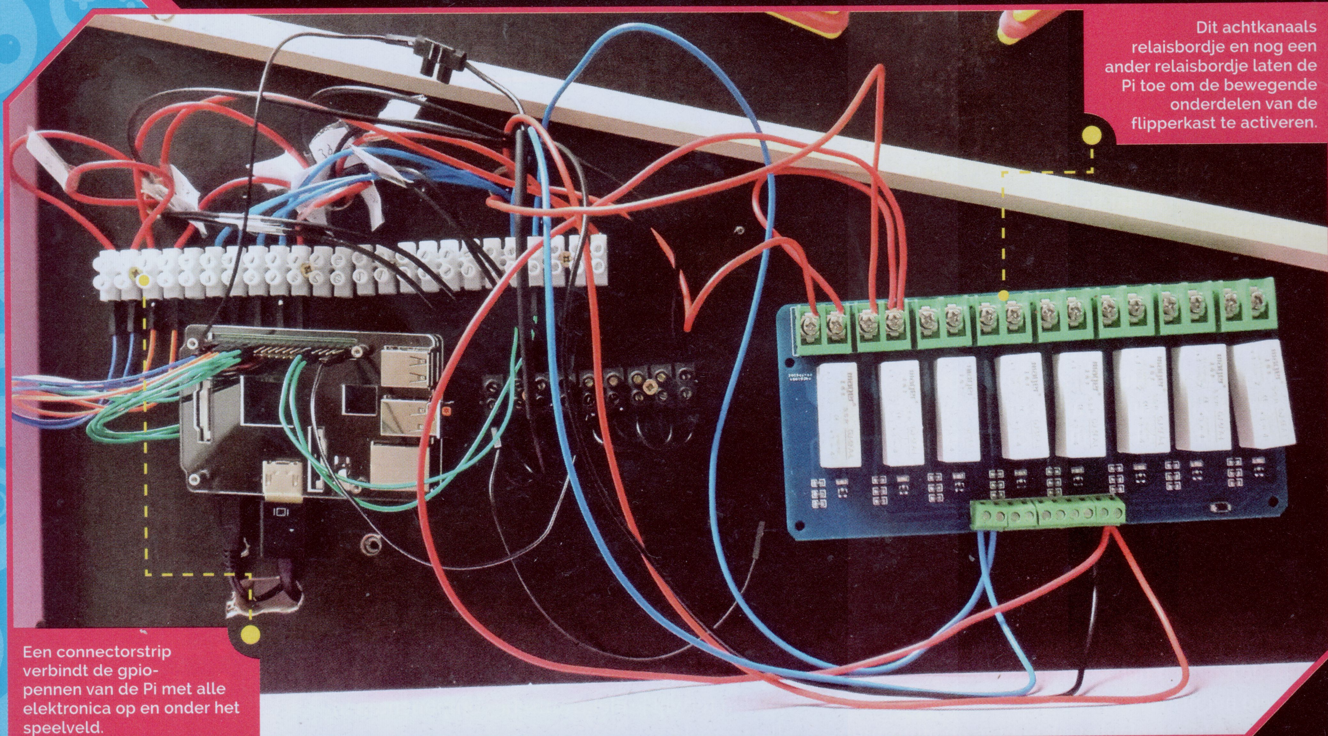
Open een terminalvenster en typ het volgende in:

```
sudo apt install python-pygame
git clone https://github.com/bishoph/
pinball.git
cd pinball
python pinball_machine.py
```

Als je op Q drukt, stopt het programma. Lettertypes zijn niet inbegrepen. Je dient eerst je eigen TrueType-bestanden **pinball.ttf** en **comicfx.ttf** te vinden (je vindt ze gratis online) en ze naar /usr/local/share/fonts/ te kopiëren.

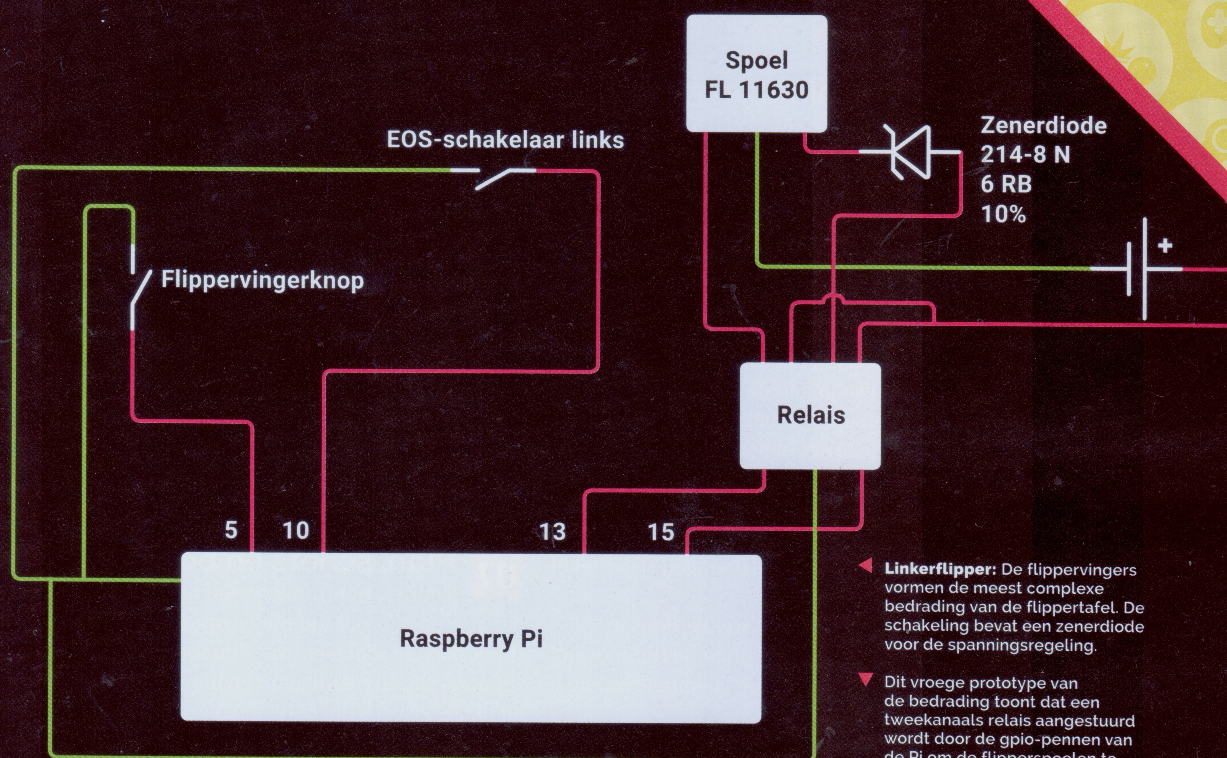
02 Beeld

Alles installeren is veel gemakkelijker als je een beeldscherm op je Pi aangesloten hebt, maar ons Python-script kan ook met een extern scherm



Dit achtkanaals relaisbordje en nog een ander relaisbordje laten de Pi toe om de bewegende onderdelen van de flipperkast te activeren.

Een connectorstrip verbindt de gpio-pennen van de Pi met alle elektronica op en onder het speelveld.

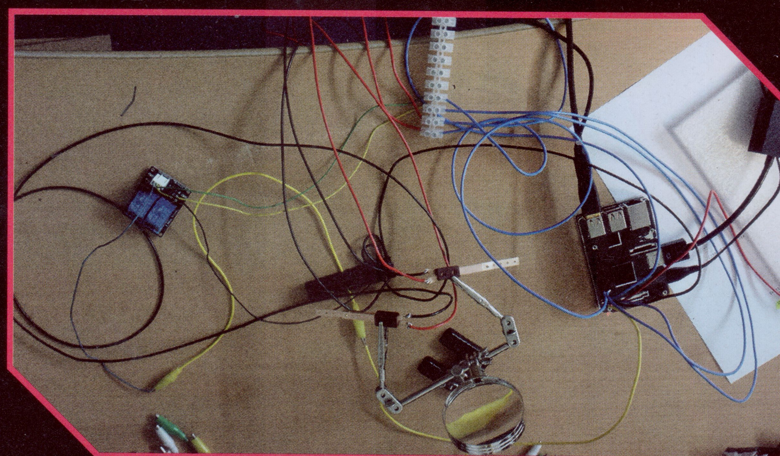


◀ **Linkerflipper:** De flippervingers vormen de meest complexe bedrading van de flippertafel. De schakeling bevat een zenerdiode voor de spanningsregeling.

▼ Dit vroege prototype van de bedrading toont dat een tweekanaals relais aangestuurd wordt door de gpio-pennen van de Pi om de flipperspoelen te activeren.

werken. We gebruiken dit om de huidige score van de speler te tonen, het aantal ballen dat je nog kunt spelen, de hoogste score van de flipperkast en wat leuke visuele effecten

Je zou de monitor ook met een VESA-beugel aan een plank of statief op de flipperkast kunnen bevestigen. Of als je met een bedframe werkt zoals Martin, bevestig de monitor dan aan wat vroeger het hoofdeinde van het bed was.



03 Muziek, maestro!

Geluid speel je ofwel door de geïntegreerde luidspreker van een beeldscherm af, ofwel door luidsprekers die je op de 3,5 mm-audiopoort van de Pi aansluit. De geluiden definieer je in het script **effects.py**, dat we in **stap 01** geïnstalleerd hebben. Je dient je eigen audiobestanden te downloaden — Martin haalde de zijne van **freesfx.co.uk**.

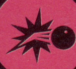
Plaats de geluidsbestanden in de directory **/home/pi/pinball/sounds** en pas **effects.py** aan. Het script speelt geluiden af wanneer je de flipperkast inschakelt, wanneer je bal door de schietbaan gaat, wanneer je bal uit het spel gaat en wanneer je bal een spinner of popbumpers activeert. En wanneer je flipperkast werkloos is, speelt het script op willekeurige momenten speciale geluiden af.

04 Frame van de flipperkast

De flipperkast van Martin Kauss begon als een kinderbed, versierd met kleurige beelden van Disney-prinsessen. Maar je kunt ook je eigen frame uit hout bouwen, een kit voor een tafelonderstel kopen of zelfs met K'nex of Meccano iets bouwen.

Het frame meet 145×77 cm. Voor het speelveld — het oppervlak van de flipperkast waar alle actie gebeurt — gebruikte Martin een multiplexplaat van 230 mm dik die hij in het zwart afwerkte.

Onder het speelveld heb je ruimte nodig voor je bedrading en voedingen.



Test eerst

Verzeker je ervan dat je componenten en verbindingen werken zoals bedoeld voordat je ze permanent op hun plaats schroeft.

05 Hoge spanning

In dit project heb je zowel een 5 V-voeding nodig, die voor de verlichting dient, als een 36 V-voeding voor de spoelen voor de flippers en bumpers, die een hogere spanning nodig hebben. Dus hoewel de spoel van de popbumper verbonden is met de 36 V-voeding, is de ingebouwde led van de popbumper aangesloten op de 5 V-voeding zoals alle andere leds in de flipperkast. Voor eenvoudige aansluitingen monteren we een extra hoge gpio-header op de gpio-header van de Pi.

Daarop passen dan gpio-kabels die naar een connectorstrip gaan. Zowel ingangen als uitgangen, en alle leds en spoelen zijn via relaisbordjes aangesloten. Die relais krijgen hun stroom via de voeding van de Raspberry Pi (via fysieke pennen 2 en 6). Ze worden elk geactiveerd door een specifieke gpio-uitgang.

Het is handig als je een kleine stekkerdoos met vier stopcontacten achteraan of onder je flipperkast bevestigt. Je hebt immers ook nog voedingen nodig voor je monitor en de Pi.

06 De juiste helling

Het speelveld van een flipperkast mag niet vlak staan, maar het kan lastig zijn om de correcte hoek te bepalen zodat de bal met de juiste snelheid naar onderen rolt. Heb je verstelbare poten onder de flipperkast geplaatst, dan kun je de helling van het speelveld eenvoudig aanpassen.

In dit geval zijn de poten gemaakt van vier vierkanten houten latten van 4.5×4.5×100 cm vooraan en 4.5×4.5×110 cm achteraan. De voorste poten zijn verstelbaar gemaakt door plastic voetjes met schroeven die je kunt verhogen of verlagen. Het vraagt wat experimenteren tijdens het opbouwen.

07 Plan, schets en boor

Zodra je het frame en een tafelblad hebt dat erin past, is het tijd om de inhoud van de flipperkast te schetsen. Meet en fotografeer de flippers, shooter, slingshot en bumpers nauwkeurig en plaats ze eens om te testen. Vergeet niet de flipperknoppen die aan de zijkanten gemonteerd komen.



▲ **Spinner:** De spinner zelf is passief, maar elke keer dat de bal de spinner omdraait, activeert die een microschakelaar, zodat de score aangepast wordt, er geluid klinkt en er leds aangaan.



▲ Voor een project zoals dit heb je een stevige selectie gereedschap, elektronische componenten en andere onderdelen nodig, evenals de ruimte om het te bouwen.

Naast deze componenten heb je nog houten en metalen strips nodig voor je banen, een gebied om de bal weer naar de schietbaan te richten en dan het gebogen bovenste deel van de flipperkast. Boor dan alle gaten die je nodig hebt om je componenten te bevestigen en te bedraden nadat je ze gemarkeerd en dubbel gecontroleerd hebt.

Vergeet niet om voldoende ruimte onderaan je flipperkast te laten voor je elektronica, inclusief de Pi!

> "Vergeet niet om voldoende ruimte onderaan je flipperkast te laten voor je elektronica"

08 Assembleer de shooter

De shooter, ook gekend als de plunger, is het eerste kritieke onderdeel van je flipperkast om te assembleren. Je kunt het geheel als een kit kopen, inclusief een stang, metalen veren, behuizing en bevestigingscomponenten zoals de externe afwerkingsplaat.

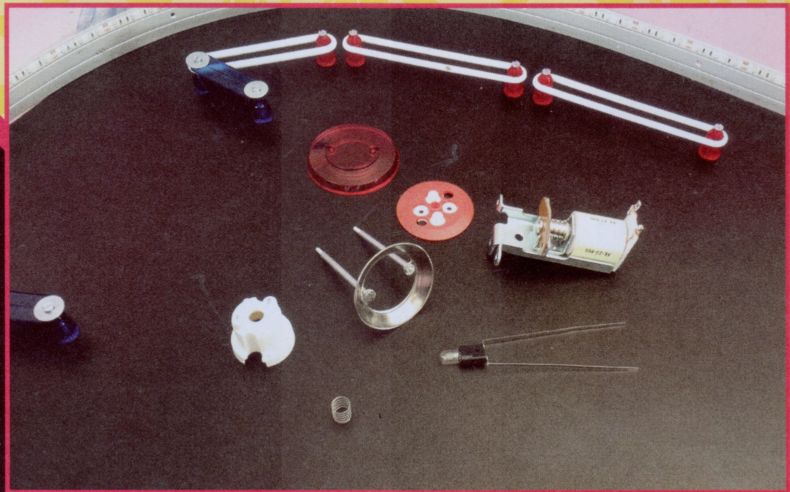
Plaats het aan de linkerkant van je flipperkast en laat de knop en de externe onderdelen uit de voorkant van je flipperkast uitsteken. Een houten lat vormt de baan waarop de bal naar beneden gaat, en het afvoerkanaal onder de flippers moet verloren ballen naar de shooter leiden. Een veer met een hoge veerspanning houdt een afsluitklep voor de baan gesloten totdat de klep opent door de snelheid van de bal die je met de shooter wegschiet.

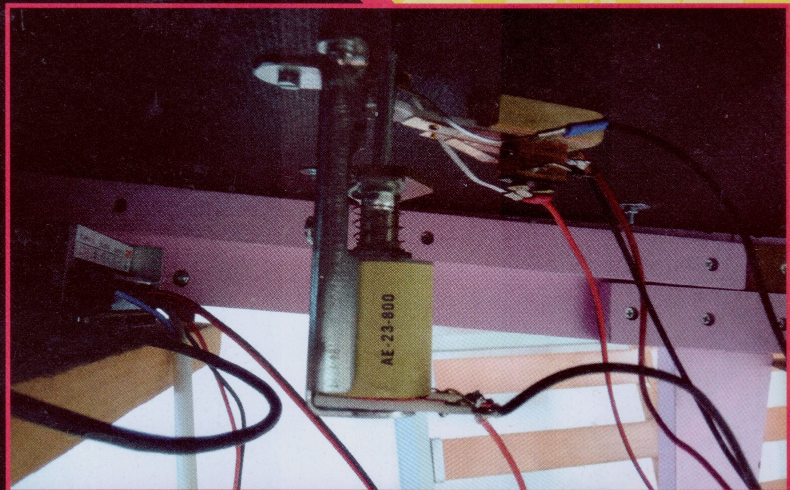
09 Rollend materieel

De meeste banen en structurele onderdelen van dit ontwerp zijn uit hout of uit gebogen aluminium strips gemaakt, bij elkaar gehouden door houten blokken die in de flipperkast geschroefd zijn. Dat maakt dit project heel vergevingsgezind als je eens een fout maakt.

Je wilt natuurlijk ook dat de Pi kan zeggen wanneer de bal door die banen stuift. Daarvoor heb je roll-over microschakelaars nodig van een leverancier van onderdelen voor flipperkasten. De microschakelaars bevestig je langs onderen aan de flipperkast. Snijd een kanaal in het speelveld met een boor en zaag. Bevestig de schakelaar op een houten strip en positioneer het onder het kanaal zodat de schakelaar genoeg uitsteekt om door het gewicht van de bal geactiveerd te worden als die passeert.

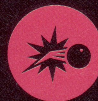
Dit project heeft ook een roll-over schakelaar aan het einde van de schietbaan en op de smalle doorgangen die de bal uit het spel halen, plus nog een schakelaar helemaal op het einde, juist voordat de bal weer de schietbaan inkomt, zodat de







▲ De popbumper wordt met zijn spoel onderaan en zijn dopje boven de tafel gemonteerd.



Meer details online

Bezoek voor extra informatie en een blog over de prinsessen-flipperkast zeker de website van Martin Kauss op. magpi.cc/iimYKq

flipperkast het weet wanneer de bal het speelveld verlaat.
Al deze schakelaars zijn met de gpio-header van de Pi verbonden — via een connectorstrip — zodat ze gebeurtenissen kunnen activeren zoals lichten, geluiden en scores.

10 Flippervingers

De flippers zijn de belangrijkste componenten van een flipperkast. Je hebt een 36 V 5 A-voeding nodig om de flipperspoelen genoeg kracht te geven, evenals een tweekanaals relais om te activeren wanneer ze door de gpio-pennen van de Pi aangestuurd worden.

Omdat we alles met een Pi aansturen, hebben we moderne flippers nodig met een *normally open* (NO) *end-of-stroke* (EOS) schakelaar.

De spoelen monteren we onderaan de flippervingers. Voor elke flipper dien je een knop aan de zijkant van de flipperkast te verbinden met een gpio-pen, de EOS-schakelaar met een andere, en dan nog twee pennen — via het relais — met de spoeleenheden, die eigenlijk twee verschillende spoeldraden hebben.

De eerste spoel, HIGH, heeft een lage weerstand en zorgt dat de flippervinger hard en snel beweegt, terwijl de tweede, HOLD, een hoge weerstand heeft en je toelaat om de flipperknoppen ingedrukt te houden om de flippers rechtop te houden.

Pen	Connector contactstrip	Type	Beschrijving	Relaisverbinding
3	1	IN	Flipperknop rechts	
5	2	IN	Flipperknop links	
8	3	IN	Flippervinger EOS rechts	
10	4	IN	Flippervinger EOS links	
7	5	IN	Microschakelaar spinner	
11	6	OUT	Flippervinger rechts HIGH	Relais #1,1
12	7	OUT	Flippervinger rechts HOLD	Relais #1,2
13	8	OUT	Flippervinger links HIGH	Relais #2,1
15	9	OUT	Flippervinger links HOLD	Relais #2,2
16	10	IN	Microschakelaar schietbaan	
18	11	IN		
19	12	IN		
21	13	IN		
22	14	IN	Schakelaar bumper #1	
23	15	IN	Schakelaar bumper #2	
2	16	IN	Schakelaar slingshot	
26	17	OUT		
29	18	OUT		
31	19	OUT		
32	20	OUT	Lamp #1, schietbaan	Relais #3,1
33	21	OUT	Lamp #2, slingshots	Relais #3,2
35	22	OUT	Lamp #3, bumper	Relais #3,3
36	23	OUT	Lamp #4, bumper	Relais #3,4
37	24	IN	Microschakelaars smalle doorgang, één signaal!	
38	25	IN	Bumper #1, spoel	Relais #4,1
40	26	IN	Bumper #2, spoel	Relais #4,2



De prinsessenflipperkast was een groot succes bij de kinderen van Martin en al hun vrienden. Ze vroegen zelfs of hij geen versie met meerdere flipperballen tegelijk kon maken.

11 In de slingshot

Slingshot bumpers zijn de spievormige componenten die juist boven de flipper staan. Ze helpen je bal om terug te kaatsen op zijn tocht rond de tafel.

Ze zijn gemaakt van *star posts* — felgekleurde plastic bevestigingen waar een rubberen ring rond past en een rubberen rand waartegen de bal kaatst. Een microschakelaarsensor detecteert wanneer de bal de bumper raakt, waarvoor je punten krijgt, en activeert dan geluiden en lichteffecten.

Wat leuke hoesjes voor de slingshots maken onze look volledig, en een baan gemaakt van hout loopt recht onder de slingshots om een route naar de flippers te bieden.

12 Een bumper of twee

Paddenstoelvormige popbumpers vormen zeker de meest iconische elementen van een flipperkast. Ze zijn te vinden in volledige kits. De meeste werken via een geïntegreerde microschakelaar, die het detecteert wanneer de bal hun plastic rand raakt. Een spoel trekt dan de staaf van de bumper naar beneden om de bal weg te schoppen.

Je kunt de basis van de bumper gebruiken om te markeren waar je ze wilt bevestigen, met de spoel onder de oppervlakte van het speelveld. De spoelen krijgen hun stroom van de 36 V-voeding en zijn met de Pi verbonden via het achtkanaals relais.

13 Een spinner

Een spinner, of ronddraaiend doel, is een klassieke component van een flipperkast met zijn eigen baan en een microschakelaar die lampen activeert en de score verhoogt wanneer de speler het doel raakt.

De baan wordt hier gemaakt van een metalen rail van een aluminium drempelstrook. De spinner wordt bevestigd met montagebeugels en bouten van de bouwmarkt, plus star posts en rubberen ringen van een leverancier van flipperkastmateriaal.

Een actuatormicroschakelaar die met de spinner verbonden is, detecteert het passeren van de bal, verhoogt dan je punten en laat de leds rond de tafel knipperen.

14 Actie!

Een flipperkast is niets zonder knipperende lichten. Dit project gebruikt een ledstrip van 2 m die verbonden is met de 5 V-voeding en verlichting biedt voor de schietbaan en de bovenste lus die helpt om de bal rond de tafel te laten rollen.

In de twee bumpers zitten er ook individuele lichten, waardoor we in totaal dus aan vier gpio-verbindingen met de Pi komen voor de verlichting. Het Python-script dat we draaien, activeert de lichten wanneer de bal op het speelveld komt, bumpers raakt, de spinner activeert of over de roll-over schakelaars passeert. De lichten worden ook op willekeurige momenten geactiveerd door het script als er geen invoer of actie is.

Opgepast met je vingers!

Raak de spoelen of bewegende onderdelen tijdens het testen niet aan. De snelle samentrekking van de solenoïde kan je vingers immers pijnlijk vastklemmen.



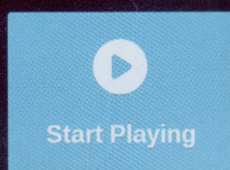


SPEEL PC-GAMES OP JE RASPBERRY PI

Maak van je Raspberry Pi een Steam Link-machine en stream games in hoge kwaliteit van Steam op je pc.

Je hebt nodig

- Raspberry Pi 3B+/3B
- Raspbian Stretch
- Toetsenbord en muis voor de installatie
- Windows-pc
- Steam (en een Steam-account)
- Compatibele controller (zie het kader achteraan dit artikel)















Voordat je games gaat spelen, controleer je het best op een desktopcomputer of je een goede verbinding hebt met Steam en of je controller goed is geconfigureerd

Klik op het icoontje **Start Playing** om Steam op je pc te starten en toegang tot je games te krijgen.

Steam is een online winkel vol met de nieuwste pc-games. Onlangs heeft de Raspberry Pi ondersteuning voor Steam Link gekregen. Met Steam Link stream je Windows-games van je pc naar je Raspberry Pi. Je hoeft alleen maar een gamepad op je Pi aan te sluiten om een spelmachine te krijgen die de concurrentie met elke spelconsole aankan.

Je kunt zowat elke gamepad met je Raspberry Pi gebruiken (zie het kader "Compatibele controllers" op het einde van dit artikel). Bovendien kun je pc-games op hoge kwaliteit op gelijk welke televisie of computermonitor in huis spelen.

Dit werkt samen met de Steam-software op een Windows-pc, dus je hebt een Steam Store-account nodig en een game uit de Steam-winkel. Die bevat overigens voldoende gratis games en demo's om te spelen.

Laten we eens kijken hoe je van je Raspberry Pi een Steam Link-machine maakt om er Windows-games op te spelen.



Lucy Hattersley

Lucy is hoofdredacteur van ons Britse zusterblad The MagPi en ze houdt van rollenspellen en avonturenspellen. Ze was echt verrast hoe leuk Euro Truck Simulator 2 is, en ze zou wel op elk moment met een virtuele vrachtwagen in Cambridge kunnen rondrijden.
magpi.cc

01 Installeer Steam

Als je nog geen Steam op je Windows-pc geïnstalleerd hebt, download het programma dan van **store.steampowered.com**. Klik op

Install Steam om het bestand **SteamSetup.exe** te downloaden. Dubbelklik erop en volg het installatieproces. Klik daarna op **Create a New Account** en volg de stappen om je account te creëren.

Start daarna Steam op en meld je aan. Je hebt een game nodig om te spelen. Gelukkig heeft Steam een geweldige selectie van gratis games en demo's om uit te proberen. Klik op **Games** en kies **Free to Play** om ze te bekijken. Wij kozen ervoor om de demo van Euro Truck Simulator 2 te installeren.

02 Installeer Steam Link

Nu Steam op je Windows-pc draait, is het tijd om van je Raspberry Pi een Steam-streamingdoosje te maken. Start met een verse installatie van Raspbian Stretch. Zorg dat je Raspberry Pi met internet verbonden is (het liefst met een ethernetverbinding) en open een Terminal-venster. Voer de volgende opdrachten uit:

```
sudo apt update
sudo apt install steamlink
```

Bevestig elke vraag met y.

03 Verbind een controller

We verbinden een PlayStation DualShock 4-gamepad met de Raspberry Pi via Bluetooth.

Je hebt de keuze uit talloze andere controllers, inclusief de Xbox One, Nintendo Switch Pro Controller en de Steam Controller — naast heel wat andere Bluetooth-gamepads. En voor veel games volstaan een toetsenbord en muis. Maar een controller is het handigste als je je Raspberry Pi met een televisie wilt gebruiken.

Hou je toetsenbord en muis met de Raspberry Pi verbonden terwijl je alles configureert. Klik op het Bluetooth-icoontje in de menubalk en kies **Turn on Bluetooth**. Klik nu opnieuw op het Bluetooth-icoontje en kies **Add Device**.

Zet de gamepad nu in koppelingsmodus. Het proces daarvoor is voor elke gamepad verschillend. Op een DualShock 4-controller houd je de PS4- en Stream-knop tegelijk ingedrukt tot het lichtje op de achterkant begint te knipperen.

In het venster **Add New Device** van je Raspberry Pi zou je nu **Wireless Controller** moeten zien. Selecteer het en klik op **Pair**. Je zou nu **Pairing Successful** moeten zien. Indien niet, probeer het dan eens met een klik op **Bluetooth > Wireless Controller > Connect**.



▲ We rijden in Cambridge rond in Euro Truck Simulator 2, op een Raspberry Pi.

Geef het niet te snel op. We hadden enkele pogingen nodig om de PS4-gamepad met de Raspberry Pi verbonden te krijgen.

04 Start Steam

Klik in het menu op **Games > Steam Link**. Je kunt het terminalvenster nu sluiten.

Steam Link opent in schermvullende modus. Je zou nu drie zaken moeten zien: de naam van je computer (in ons geval **DESKTOPE1QNN21**), de naam van je controller (bij ons **PS4 Controller**) en **Good Connection**.

Laten we eerst de verbinding testen. Klik op **Settings** met het gamepad en kies dan **Streaming and Network test** om het venster met de netwerktest te laten verschijnen. Hopelijk zie je **Network Test Complete** en **Fantastic! Your network looks like it will work well with Steam Link**. Steam raadt een ethernetverbinding aan, maar als je wifi wilt gebruiken, kun je de prestaties verbeteren door je Pi dichter bij je draadloos toegangspunt te plaatsen. Klik op **OK** en keer met het icoontje linksboven terug naar het hoofdscherm.

05 Start met spelen

Klik op het icoontje **Start Playing** in Steam Link. Het programma zou je nu een pincode van vier cijfers moeten tonen. Voer die pincode in het dialoogvenster **Authorize Device** van Steam op je Windows-pc in en klik op **OK**.

Op een gegeven moment zie je misschien een waarschuwing **Can't connect** met een boodschap dat je voor streaming extra drivers dient te installeren. Klik dan op **Install** om de benodigde drivers te downloaden en te installeren en klik daarna opnieuw op **Start Playing**.

Ethernet-verbinding

Steam raadt je aan om een bedrade ethernetverbinding te gebruiken tussen je Raspberry Pi en je router. We vonden dat games streamen over wifi prima ging, maar sommige menu's waren een beetje traag. Hopelijk verbeteren de prestaties mettertijd nog.



06 Welkom in Steam

Je ziet nu het venster **Welcome to Steam** in Steam Link. Bovenaan zie je een strook met icoontjes: **Web, Store, Library, Community** en **Chat**. Daarboven zie je drie kleinere icoontjes voor downloads, instellingen en een aan-uitknop. Met de D-pad op het PS4-gamepad zou je nu door de icoontjes moeten kunnen navigeren. Druk op de X-knop om een icoontje te selecteren en O om terug te keren.

07 Instellingen van de controller

De basisondersteuning voor de gamepad werkt, maar voor betere ondersteuning dien je de controller in te stellen. Klik op het icoontje van het tandwiel bovenaan rechts en kies **Controller Settings**. Ga naar **PlayStation Configuration Support** en vink het vakje aan. Het venster **Personalise Your Controller** verschijnt nu. Laat de standaardinstellingen voorlopig staan. Kies **Submit** en druk op O om naar het hoofdscherm terug te keren.

08 Bekijk je bibliotheek

Selecteer **Library** in het venster **Welcome to Steam** om alle games in je Steam-collectie te bekijken. Kies **Installed** en dan **Euro Truck Simulator 2 Demo**. Je ziet nu een gele waarschuwing **Controller Configuration Required**. Hoewel je net je PS4-gamepad in Steam ingesteld hebt, vereist het mogelijk nog extra configuratie voor individuele games — heel wat Steam-games zijn immers ontworpen voor een volledig toetsenbord en muis. We komen hier dadelijk op terug; klik nu gewoon op **Play**.

09 Configureer de controller

Omdat dit de eerste keer is dat je het spel start, verschijnt de **First Time Config Selection**. Klik op **Browse Other Configs**. Kies **Community**, waarna je een boel toetsenkoppelingen ziet die door anderen aangemaakt zijn. Rechts ervan zie je het aantal stemmen. Kies degene met de meeste stemmen. We zien hier **zaka.ahoa2's Bindings** met vier stemmen. Druk op de knop met het vierkantje op je gamepad om de configuratie toe te passen en start het spel op.

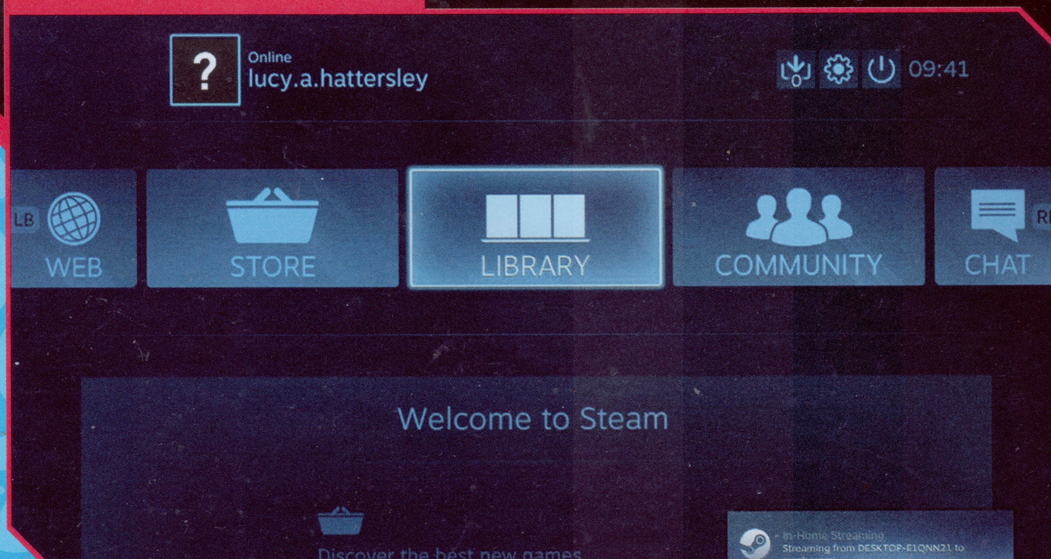
10 Stel je eigen besturing in

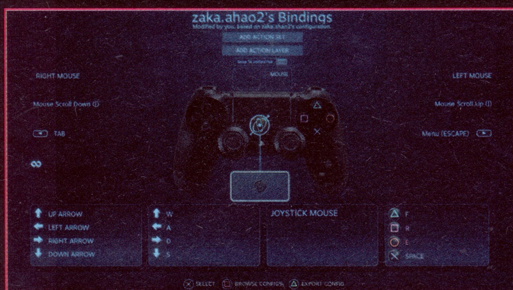
Je kunt in het spel nu beginnen rondrijden met de PS4-trackpad. De D-pad of de linkse analoge stick bestuurt je bewegingen, en met de rechtse analoge stick kun je rondkijken. Maak maar eens een ritje (of speel gelijk welk game dat je gekozen hebt). Het is het best om met toetsenkoppelingen van de community te starten, maar je kunt je eigen besturing toevoegen aan elk game en ze naar je eigen voorkeuren personaliseren.

Forum van Steam Link

Als je vragen hebt, krijg je zeker hulp op het forum van Steam Link. Er is zelfs een hele categorie over problemen oplossen op de Raspberry Pi. **magpi.cc/xJopzO**

▼ De interface van Steam is speciaal ontworpen voor gebruik op een televisiescherm met een controller.




▲ Zelfs als je controller met Steam Link verbonden is, krijg je in veel games waarschuwingen over de configuratie.

11 Aangepaste besturing

Druk op de PlayStation-knop voor het venster met instellingen. Je krijgt hier de controllerconfiguratie te zien, en hier kun je aangepaste controllers instellen. Selecteer de linkse schouderknop en gebruik het virtuele toetsenbord om het op [in te stellen. Doe dan hetzelfde met de rechtse schouderknop voor]. Daarna kun je tijdens je rit met de linkse en rechtse schouderknoppen je richtingaanwijzers inschakelen.

Heel wat pc-games hebben een complexe besturing. Je zult dan ook niet alle games met alleen maar een gamepad kunnen spelen. Maar veel games zijn met een beetje configuratie perfect te spelen. Veel plezier met de pc-games op je Raspberry Pi!

12 Speel het game

Speel nu je game maar. Zorg dat alle besturingsknoppen correct werken. Daarna kun je je Raspberry Pi bij je televisie plaatsen. Om nu Steam Link automatisch uit te voeren als je je Pi opstart, open je het autostart-bestand van de grafische sessie van Raspbian met de volgende opdracht in Raspbian:

```
sudo nano /etc/xdg/lxsession/LXDE-pi/
autostart
```

Voeg daar helemaal achteraan de volgende regel toe:

```
/usr/bin/steamlink %U
```

Druk op **Ctrl+O** om het bestand op te slaan en **Ctrl+X** om nano te verlaten. Voer dan het configuratieprogramma van Raspbian uit met:

```
sudo raspi-config
```

Open **3 Boot Options**, daarna **B1 Desktop / CLI** en kies daar **B4 Desktop Autologin**. Sluit het programma af en volg de suggestie om te rebooten om te testen of Steam Link automatisch opstart.

COMPATIBELE CONTROLLERS

Met deze gamepads speel je eenvoudig je games. Een volledige lijst van ondersteunde adapters en functies vind je op **magpi.cc/BoqAxE**

PS4 DualShock 4



De PS4 DualShock 4 Controller is een populaire keuze, en het gamepad dat we in dit artikel gebruiken. Het heeft niet alleen een D-pad en twee analoge sticks, maar ook een touchpad om muisinvoer na te bootsen.

Xbox-controller



De Xbox-controller werkt draadloos met dezelfde configuratie als de PS4-controller. Hoewel het niet zo goed ondersteund is als de DualShock 4, kun je er zeker heel wat games mee spelen.

Nintendo Switch Pro Controller



De Nintendo Switch Pro Controller is een duurdere optie en heeft niet de touchpad van de DualShock 4. Maar als je er nog een hebt liggen, werkt hij goed met Steam Link.

Steam Controller

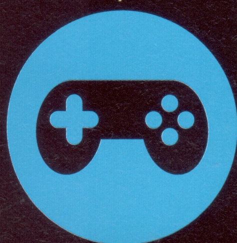


Steam verkocht ook zijn eigen unieke controller samen met de officiële Steam Machines (die de Raspberry Pi nu vervangt). Je vindt ze nog altijd en ze zijn goedkoper dan de andere controllers. Naast een analoge stick zijn er ook twee trackpads voor muisinvoer. **magpi.cc/oBzbjt**



Deel 01

PROGRAMMEER EEN ISOMETRISCH AVONTURENSPEL AMAZEBALLS



Pygame Zero in 3d? Ja, dat is mogelijk! In deze reeks van drie artikels tonen we je hoe je een 3d-spel maakt.

In onze reeks over Pygame Zero heb je tot nu toe geleerd hoe je snel games creëert. In dit eerste deel van drie artikels gebruiken we nog meer nieuwe technieken om een spel te creëren dat zich in een 3d-doolhof afspeelt. De 3d-stijl die we gebruiken, wordt isometrisch genoemd. Dat betekent dat alles wat we op het scherm tonen uit regelmatige kubussen bestaat die een enigszins 'verkeerd' perspectief hebben omdat het eigenlijk uit 2d-afbeeldingen bestaat. Deze techniek is in talloze games uit de jaren 1980 gebruikt, zoals Knight Lore, Alien 8 en mijn eigen serie, ArcVenture. In dit eerste deel bouwen we de basiskaart op met datalijsten en creëren we een stuiterbal als spelfiguur waarmee de speler in het doolhof beweegt.

01 Welkom in de derde dimensie

Pygame Zero was niet geschreven met 3d-games in het achterhoofd, maar soms dien je de grenzen een beetje te verleggen en te kijken hoe ver je met een idee kunt gaan. Onze manier om het speelveld te tekenen is strikt gezien niet driedimensionaal, maar het lijkt wel wat 3d en met deze techniek is er heel wat mogelijk. We gaan een kaart maken uit een lijst gegevens en bouwen deze uit kubussen op. Daarna voegen we een stuiterbal aan het speelveld toe, die de speler met het toetsenbord beweegt. We starten met de bal aan één zijde van het doolhof en wanneer de speler hem aan de andere zijde heeft gebracht, is het spel voltooid.

02 Kaartenmaker

Zoals vaak in deze reeks starten we met de module `pgzrun` te importeren. Vergeet niet om `pgzrun()` op het einde van je code aan te



Mark Vanstone

Mark was in de jaren 1990 ontwikkelaar van educatieve software en auteur van de ArcVenture-serie, maar verdween daarna in de wereld van de bedrijfssoftware. De Raspberry Pi heeft hem daaruit gered!

magpi.cc/YiZnxL | @mindexplorers

De bal start aan één zijde van het doolhof en de speler moet hem naar de andere zijde leiden.

De speler wordt voorgesteld door een stuiterbal.

De muren van het doolhof creëren we door kubussen te tekenen.



roepen. De standaard venstergrootte zou voor ons doel geschikt moeten zijn. We gaan nu een tweedimensionale lijst van getallen aanmaken die onze kaart voorstelt. Elk getal stelt een vakje op de kaart voor en we maken onze kaart twaalf vakjes breed en twaalf hoog. Bekijk in **figure1.py** hoe we deze lijst definiëren, die we `mapData` noemen. De andere variabele `mapInfo` definieert de breedte en hoogte van onze kaart in een structuur die in Python een *dictionary* heet.

03 A is voor aardvarken

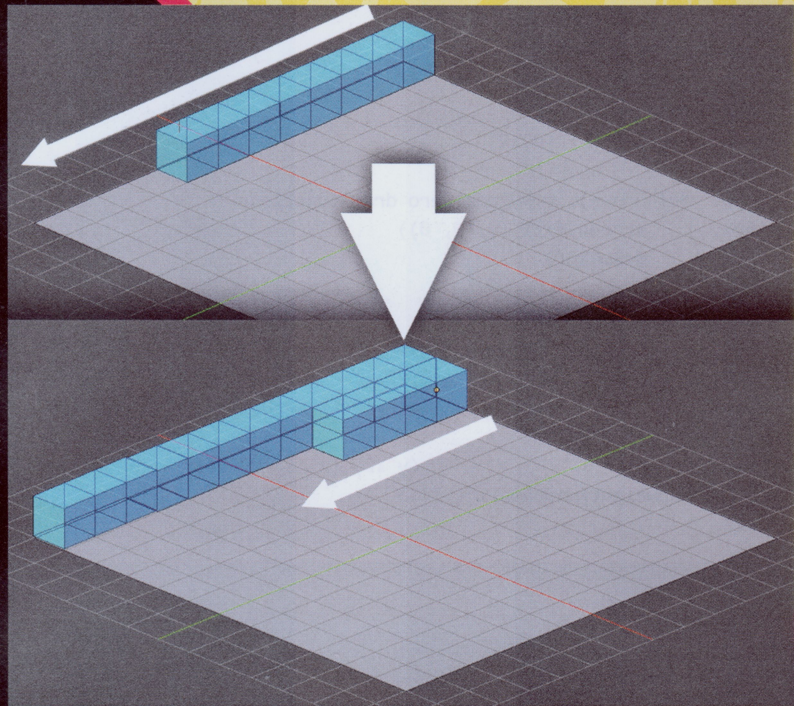
Een dictionary in Python is een heel nuttige datastructuur. In plaats van gewoon een lijst van waardes of strings op te slaan, geven we elk van deze waardes een naam. Als je de dictionary `mapInfo` eens bekijkt, zie je dat die twee waardes declareert: de eerste met de naam **width** en de tweede met de naam **height**. De eerste waarde lezen we eenvoudigweg uit met `mapInfo["width"]`, die ons in dit geval de waarde 12 zou teruggeven. Dictionary's zijn een hele nuttige structuur om meerdere variabelen die met elkaar te maken hebben samen op te slaan.

04 Breng de kaart in kaart

Nu we wat gegevens voor onze kaart hebben, dienen we afbeeldingen te definiëren waarmee we de kaart tekenen. We kunnen een nieuwe lijst van blokken voor de kaart aanmaken met `mapBlocks = ["map1c", "map2c"]`. Hiermee definiëren we: als we een 0 in de lijst `mapData` zien, teken dan de eerste afbeelding in de lijst, **map1c**, en als we een 1 in de lijst mapData zien, gebruik dan de afbeelding **map2c**. Voorlopig houden we het bij twee verschillende blokken voor de kaart. Het eerste blok stelt de vloer voor, het tweede de muren.

05 Toon me de kaart

De volgende stap is dat we onze kaartgegevens omzetten in een kaart die we op het scherm kunnen zien. De basis zetten we op in de functie `draw()` van Pygame Zero en daarna roepen we een functie `drawMap()` aan die we schrijven om het echte werk te doen. In **figure2.py** zie je dat we het scherm eerst met zwart vullen en dan onze kaart tekenen. De functie `drawMap()`, hoewel kort, ziet er misschien wat ingewikkeld uit. Laten we er daarom eens stap voor stap door gaan om te begrijpen wat erin gebeurt.



▲ We plaatsen de blokken met hun zijden tegen elkaar, één rij per keer, om een 3d-effect op te leveren.

▼ De basis van ons Pygame Zero-framewrk en de definitie van onze kaart.

figure1.py

› Taal: **Python**

DOWNLOAD DE VOLLEDIGE CODE:



magpi.cc/DVNCvv

```python
import pgzrun

mapData = [[1,1,1,0,1,1,1,1,1,1,1,1],
           [1,0,0,0,0,0,0,0,0,0,0,1],
           [1,1,1,1,1,1,1,0,1,1,1,1],
           [1,0,0,0,0,0,0,0,0,0,0,1],
           [1,1,1,1,1,1,1,1,0,0,0,1],
           [1,0,0,0,0,0,0,1,0,1,1,1],
           [1,0,1,0,1,1,0,1,0,0,0,1],
           [1,0,1,0,1,0,0,1,1,1,0,1],
           [1,0,1,0,1,0,0,0,0,0,0,1],
           [1,1,1,0,1,1,1,1,1,1,1,1],
           [1,0,0,0,0,0,0,0,0,0,0,1],
           [1,1,1,1,1,1,1,1,0,1,1,1]]

mapInfo = {"width":12, "height":12}

pgzrun.go()

```



figure2.py

```
OFFSETX = 368
OFFSETY = 200
mapBlocks = ["map1c","map2c"]
mapHeight = [0,32]

def draw(): # Pygame Zero draw function
    screen.fill((0, 0, 0))
    drawMap()

def drawMap():
    for x in range(0, 12):
        for y in range(0, 12):
            screen.blit(mapBlocks[mapData[x][y]], ((x*32)-(y*32)+OFFSETX,
                              (y*16)+(x*16)+OFFSETY - mapHeight[mapData[x][y]]))
```

▲ De functie waarmee we de kaart tekenen is kort maar ziet er ingewikkeld uit.

▼ Deze techniek werd gebruikt door talloze 8-bits games en zelfs de vroege versies van SimCity en Age of Empires maakten er nog gebruik van.

06 Bouwstenen

In een geneste lus gaan we door onze gegevens en tonen dan de kaart. De eerste lus is voor de x-richting, en daarin hebben we een andere lus voor de y-richting. De x en y verwijzen in dit geval niet naar coördinaten van schermpixels, maar naar onze datablokken. Beeld je nu in dat we onze data met 45 graden draaien, zodat we onze blokken diagonaal op het scherm tekenen. We doen dat met een beetje wiskunde om de x- en y-posities in onze gegevens te vertalen naar locaties op het scherm.

07 Een prachtig uitzicht

Om elk blok te tekenen, gebruiken we de functie `screen.blit()` van Pygame Zero, die een afbeelding inlaadt en ze op de gespecificeerde coördinaten op het scherm tekent. De eerste parameter van deze functie is dus de naam van de afbeelding. Die verkrijgen we door de waarde uit `mapData[x][y]` te halen en met `mapBlocks` de naam van de bijbehorende afbeelding terug te geven. Voor de tweede parameter van de functie berekenen we de schermcoördinaten. Elk blok is 64 pixels breed, maar omdat we ze diagonaal tonen, dienen we voor

elk blok zijwaarts 32 pixels te verschuiven zodat de blokken overlappen. In de verticale richting bewegen we voor elk blok 16 pixels omlaag.

08 Diamantvorm

Om onze x-coördinaat op het scherm te verkrijgen, vermenigvuldigen we de x-waarde van de data met 32 en trekken daar de y-waarde maal 32 van af. Dat geeft ons een schermcoördinaat die begint van 0. Daar tellen we dan nog een vooraf gedefinieerde beginwaarde bij op om het eerste blok naar het midden van het scherm te verplaatsen. We doen hetzelfde met de y-coördinaat, maar vermenigvuldigen de x- en y-waardes zoals gezegd met 16, tellen een beginwaarde op zodat het eerste blok niet helemaal bovenaan het scherm begint en houden er met `mapHeight` rekening mee dat sommige blokken hoger zijn dan andere. Wanneer we deze code uitvoeren, zien we twaalf rijen en twaalf kolommen in een diamantvorm getekend worden.

> "We creëren een dictionary die alle gegevens bevat die we over de speler moeten weten"

09 Stuiterbal

Dan is het nu tijd om onze spelfiguur te maken, die we in dit geval door een stuiterbal voorstellen. Laten we beginnen met de bal eerst te positioneren en op de kaart te laten bewegen. We creëren een dictionary die alle gegevens bevat die we over de speler moeten weten. Zie in **figure3.py** hoe we die spelergegevens definiëren. De `x`- en `y`-waardes definiëren waar de speler zich bevindt in de kaart met blokgegevens. Zoals je in de dictionary ziet, start de speler in kolom (`x`) 0 en rij (`y`) 3. De waarde frame vertelt ons welk animatieframe we dienen te tonen. En de waardes `sx` en `sy` zijn de eigenlijke schermcoördinaten waar we de bal tekenen.

10 Teken de bal op het scherm

Vergeet voorlopig even de andere waardes in de dictionary voor onze speler. Bekijk dan eens het andere stukje code in **figure3.py**. Dat komt in onze lussen in `drawMap()`, vlak nadat we de blokken tekenen met `screen.blit`. De code zegt: "Als het blok dat we momenteel aan het afhandelen zijn het blok is waarop de speler zich bevindt, bepaal dan de schermcoördinaten (met dezelfde berekeningen als de blokken) en teken de bal op het scherm." We bepalen die schermcoördinaten van de bal maar één keer, omdat we nu een functie `doMove()` maken die de schermcoördinaten van de speler afhandelt.

11 Onderweg

Onze functie `doMove()` introduceert enkele nieuwe Python-technieken en nog wat meer van de gegevens in de dictionary van de speler. We geven drie parameters aan `doMove()` door. De eerste is de dictionarystructuur voor de speler (zodat we de spelerwaardes kunnen lezen en schrijven), en dan de veranderingen in x en y die we willen maken in blokeenheden. De `x` en `y` zijn 0, 1 of -1 en stellen een beweging in onze blokgegevens van de kaart voor. Het eerste deel van `doMove()` controleert of het blok waarnaar we willen bewegen zich binnen de grenzen van onze kaart bevindt. Dat is een verkorte manier om meerdere waardes te vergelijken. In eenvoudige termen is het `0 <= x < width`, wat betekent dat x tussen 0 en de breedte -1 moet liggen. Zie **figure4.py** voor de code die we gebruiken.

Gegevens-opmaak

Als je handgecodeerde gegevens gebruikt, is het een goed idee om ze op zo'n manier op te maken dat je ze eenvoudig kunt lezen.

▼ We definiëren de datastructuur voor de speler en tekenen de speler op het scherm.

figure3.py

```
# This code is near the top of our program

player = {"x":0, "y":3, "frame":0, "sx":0, "sy":0,
          "moveX":0, "moveY":0, "queueX":0, "queueY":0,
          "moveDone":True, "movingNow":False,
"animCounter":0}


# This code goes in the drawMap() function inside the y loop

            if x == player["x"] and y == player["y"]:
                if player["sx"] == 0:
                    player["sx"] = (x*32)-(y*32)+OFFSETX
                    player["sy"] = (y*16)+(x*16)+OFFSETY-32
                screen.blit("ball"+str(player["frame"]),
(player["sx"], player["sy"]))


```



figure4.py

```
def update(): # Pygame Zero update function
    global player
    if player["moveDone"] == True:
        if keyboard.left:
            doMove(player, -1, 0)
        if keyboard.right:
            doMove(player, 1, 0)
        if keyboard.up:
            doMove(player, 0, -1)
        if keyboard.down:
            doMove(player, 0, 1)
    updateBall(player)

def doMove(p, x, y):
    if 0 <= (p["x"]+x) < mapInfo["width"] and 0 <=
(p["y"]+y) < mapInfo["height"]:
        if mapData[p["x"]+x][p["y"]+y] == 0:
            p.update({"queueX":x, "queueY":y,
"moveDone":False})
```

▲ We lezen de toetsenbordinvoer in en antwoorden erop door de data in te stellen om de spelfiguur te bewegen.

12 Watch your step

Nadat we hebben gecontroleerd of de speler door zijn beweging in het kaartgebied blijft, kunnen we testen of de beweging naar een vloerblok leidt (waarde 0 in onze gegevens). We dienen alleen maar de waarde in **mapData** te controleren. De volgende regel code is een slimme manier om meerdere waardes tegelijk in een dictionary te veranderen. Met de functie **p.update()** zetten we de waardes van **queueX**, **queueY** en **moveDone** allemaal tegelijk. De reden waarom we de beweging in een wachtrij (*queue*) plaatsen in plaats van onmiddellijk te bewegen is omdat we misschien al in beweging zijn en we dan willen wachten tot het einde van de huidige beweging.

> “ De reden waarom we de beweging in een wachtrij plaatsen is omdat we misschien al in beweging zijn ”

Een dictionary

Als je gegevens in een dictionary opslaat, maak je het eenvoudiger om te begrijpen waarvoor je de gegevens gebruikt. Het is ook een goed opstapje naar objectgeoriënteerd programmeren (OOP).

13 De update

We zitten nog altijd in **figure4.py**. Daar zie je hoe we de bewegingen aansturen met toetsenbordinvoer. In de functie **update()** van Pygame Zero kijken we of de cursortoetsen ingedrukt zijn. Indien ja, dan roepen we onze functie **doMove()** aan met geschikte parameters voor de beweging. Daarna roepen we een functie **updateBall()** aan, die al het zware werk van de animatie van de bal en de beweging van één blok naar het volgende uitvoert. Merk op: voordat we op toetsindrukken controleren, zorgen we dat we klaar zijn voor meer invoer door in de dictionary van de speler de waarde **moveDone** te controleren. Die zetten we op **False** in **doMove()**.

14 Frames van de animatie

Al wat we nu nog moeten doen, is de bal van één blok naar het volgende laten bewegen. Maar als we dat in de verkeerde volgorde doen, eindigen we met de bal die vóór blokken getekend wordt terwijl hij erachter hoort, of achter blokken terwijl hij ervoor hoort. In deze fase kunnen we de frames van de animatie introduceren om de bal op en neer te doen stuiteren. We willen ook dat de bal vloeiend beweegt van de ene plaats naar een andere. Dus hoe kleiner de beweging van één **draw()** naar de volgende, hoe beter.

15 Animatieframes

Laten we eens starten met de animatieframes voor de stuiterende bal. We hebben acht frames met de bestandsnamen **ball0.png** tot en met **ball7.png**. Als we gewoon de waarde frame in de dictionary van de speler verhogen telkens we **updateBall()** aanroepen en de waarde weer op 0 zetten als we aan 8 komen, zal de functie **drawMap()** de animatie in een lus tekenen. Het enige probleem: als we de animatie op deze snelheid afspelen, stuitert de bal heel snel. We dienen hem dus te vertragen. Daarvoor gebruiken we een andere waarde uit de dictionary van de speler, **animCounter**. In deze waarde tellen we elke vier frames. Op de vierde frame verhogen we **frame** met 1.

16 Beeld voor beeld

De timing van de beweging van de bal moet overeenkomen met de correcte frames, zodat het lijkt alsof hij vloeiend stuitert tijdens de beweging.

We willen tot `frame = 4` wachten voor we met de beweging beginnen. Op dat moment geven we de waardes `queueX` en `queueY` door aan `moveX` en `moveY`, en we zetten `movingNow` op `True`. Als `movingNow` gelijk aan `True` is, worden de waardes `sx` en `sy` van de speler veranderd. Voor elk blok dienen we 32 pixels naar links of rechts te bewegen in stappen van 1 pixel, en 16 pixels omhoog of omlaag in stappen van een halve pixel. Onze beweging heeft dus 32 updatecycli nodig.

17 Van één blok naar het volgende

Om onze bewegende bal in de juiste volgorde op de kaart te tonen, dienen we het blok waarop hij staat te veranderen in het correcte frame van de animatie. Op het moment dat `frame = 7`, is het tijd om te veranderen. Op dat punt werken we de waardes `x` en `y` van de speler bij, gebaseerd op de waardes van `moveX` en `moveY`. Daarna zetten we moveDone op True. Dat betekent dat we wanneer we terugkeren naar `frame = 4` de waardes `moveX` en `moveY` weer op `0` kunnen zetten, en `movingNow` op `False` tenzij er ondertussen een andere beweging in de wachtrij staat.

18 Alles samen

In **figure5.py** zie je in de functie `updateBall()` hoe deze opeenvolging van frames werkt. Je ziet de controle op `movingNow` eerst en het gebruik van een afzonderlijke functie, `moveP()`, om de schermcoördinaten van de speler te veranderen. Daarna komt alle logica voor de opeenvolging van acties in het huidige frame. Je ziet dat we ook controleren of het doolhof opgelost is. We zetten in het begin een globale variabele `mazeSolved` op `False`. Als de speler op het blok met coördinaten (11, 8) arriveert, zetten we de variabele op `True` en tonen we een boodschap in `draw()`.

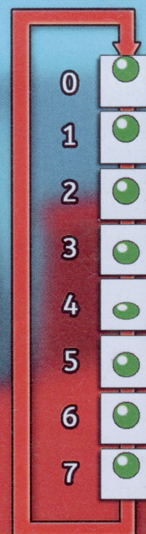
19 Level 1 uitgespeeld

Nou, dat is het eerste deel van deze reeks. Je hebt gezien hoe je uit data en 2d-afbeeldingen een kaart creëert die er als 3d uitziet, en hoe je een geanimeerde spelfiguur rond de kaart beweegt. Dit spelformaat heeft talloze mogelijkheden. In het volgende artikel kijken we hoe we de kaart groter maken, hoe we de kaart in een externe editor bewerken en hoe we de data uit een afzonderlijk bestand inladen.

figure5.py

```
def updateBall(p):
    global mazeSolved
    if p["movingNow"]:
        if p["moveX"] == -1: moveP(p,-1,-0.5)
        if p["moveX"] == 1: moveP(p,1,0.5)
        if p["moveY"] == -1: moveP(p,1,-0.5)
        if p["moveY"] == 1: moveP(p,-1,0.5)
    p["animCounter"] += 1
    if p["animCounter"] == 4:
        p["animCounter"] = 0
        p["frame"] += 1
        if p["frame"] > 7:
            p["frame"] = 0
        if p["frame"] == 4:
            if p["moveDone"] == False:
                if p["queueX"] != 0 or p["queueY"] !=0:
                    p.update({"moveX":p["queueX"],
"moveY":p["queueY"], "queueX":0, "queueY":0,
"movingNow": True})
            else:
                p.update({"moveX":0, "moveY":0,
"movingNow":False})
                if p["x"] == 11 and p["y"] == 8:
                    mazeSolved = True

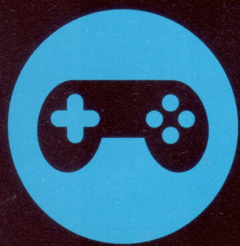
        if p["frame"] == 7 and p["moveDone"] == False and
p["movingNow"] == True:
            p["x"] += p["moveX"]
            p["y"] += p["moveY"]
            p["moveDone"] = True
```





Deel 02

PROGRAMMEER EEN ISOMETRISCH AVONTURENSPEL AMAZEBALLS



In het vorige artikel programmeerden we Pygame Zero in 3d. Laten we in deel 2 de kaart eens groter maken.

Je hebt nodig

- Raspbian Jessie of nieuwer
- De gratis kaartbewerker Tiled **mapeditor.org**
- Een beeldverwerkingsprogramma zoals GIMP of afbeeldingen van **magpi.cc/fPBrhM**
- Pygame Zero 1.2

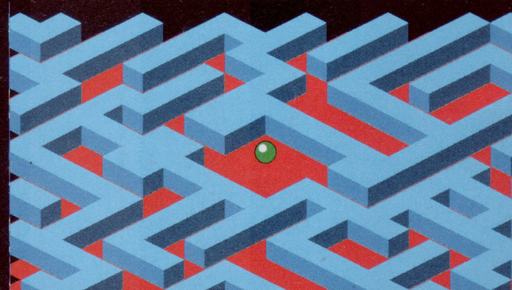
In dit tweede deel van onze reeks waarin we een 3d isometrisch game in Pygame Zero ontwikkelen, starten we vanaf waar we in het eerste artikel gebleven waren. We gaan ons 3d-gebied groter maken en kijken hoe we het gemakkelijker kunnen aanpassen. We doen dat met een kaartbewerker (map editor), Tiled, een gratis te gebruiken programma om je eigen 3d-kaarten te maken. Je leert op de volgende pagina's hoe je een eenvoudige kaart in Tiled maakt en het exporteert naar het json-formaat. Dan importeren we de kaart in ons spel en herschrijven we de functie draw zodat die rond de map scrollt wanneer de speler beweegt. We hebben heel wat te doen, dus laten we onmiddellijk beginnen.

01 Het juiste gereedschap

In het eerste deel van deze reeks creëerden we onze kaartgegevens door een tweedimensionale lijst van nullen en enen op te stellen die een vloerblok of muurblok voorstelden. De speler was in staat om naar elk blok te verplaatsen dat door een nul voorgesteld werd. Deze keer laten we het werk doen door een kaartbewerkingsprogramma in plaats van zelf de gegevens in te typen. We kiezen daarvoor Tiled. De homepage van het programma vind je op **mapeditor.org**. Als je na dit artikel het programma leuk vindt, dan vind je hier mogelijkheden om de ontwikkelaar te ondersteunen.

02 Betegeld

Tiled draait op diverse systemen, inclusief pc's, Macs en — belangrijker voor ons — op de Raspberry Pi in Raspbian. Open daarvoor een terminalvenster en update de pakketlijsten eerst met `sudo apt update`. Upgrade de pakketten naar hun nieuwste versie met `sudo apt upgrade` en installeer daarna Tiled met `sudo apt install tiled`. In het Graphics-submenu verschijnt daarna een icoontje `Tiled`.



▲ Wanneer we onze functie `drawMap()` herschreven hebben, zien we gekartelde randen aan de uiteindes van het tekengebied.



Mark Vanstone

Mark was in de jaren 1990 ontwikkelaar van educatieve software en auteur van de ArcVenture-serie, maar verdween daarna in de wereld van de bedrijfssoftware. De Raspberry Pi heeft hem daaruit gered!

magpi.cc/YiZnxL | @mindexplorers

03 Cartografie

Voorheen was onze kaart 12 bij 12 blokken groot, wat allemaal op het scherm paste. Met onze kaartbewerker kunnen we een veel grotere kaart maken. Ze kan gigantisch zijn, maar laten we het voorlopig op 30 bij 30 blokken houden. Onze kant en klare kaart en blokken kun je downloaden van **magpi.cc/fPBrhM**. Als je de kaart inlaadt, zou je een doolhof in blauw en rood moeten zien, een beetje zoals in het vorige artikel maar dan veel groter. Deze keer voegden we ook een nieuw blok toe om het eindpunt van het doolhof aan te duiden. Je zou moeten kunnen rondscrollen om de hele kaart te zien.

04 Exporteer de gegevens

Speel eens wat met de kaartbewerker. Op de website vind je geweldige documentatie. Wanneer je wat vertrouwd bent met de werking, kun je beginnen denken aan hoe je de kaartgegevens in je spel krijgt. Om de data te exporteren, ga je in het menu `File` naar `Export`. Wanneer het dialoogvenster verschijnt en `export as…` vraagt, zoek je een geschikte locatie (bijvoorbeeld een subdirectory **maps**) om de kaart in op te slaan (bijvoorbeeld als **map1**). Sla het bestand echter nog niet op, maar selecteer eerst in het uitklapmenu `Save as type` het bestandstype **Json map files** (*.json). Dit type bestand (spreek het uit als het Engelse 'Jay-son') staat voor JavaScript Object Notation en je kunt het in een teksteditor bekijken.

05 Json en de Argonauten

Als je het json-bestand opent, zie je een boel gekrulde en vierkanten haakjes met woorden en getallen over je hele scherm verspreid. Voordat je uitroept dat het allemaal Chinees voor je is, laten we je even stilstaan bij enkele van de elementen, zodat je de structuur van de data begrijpt. Als je vertrouwd bent met de programmeertaal JavaScript, herken je zeker dat de gekrulde haakjes { en } gebruikt worden om blokken code of data te omvatten, en vierkante haakjes [en] voor lijsten van gegevens. Als je het element **layers** bekijkt, zie je de data die je kaart beschrijven.

06 Laad de gegevens in

Voor dit spel hebben we eigenlijk niet alle gegevens nodig die het json-bestand bevat, maar

figure1.py

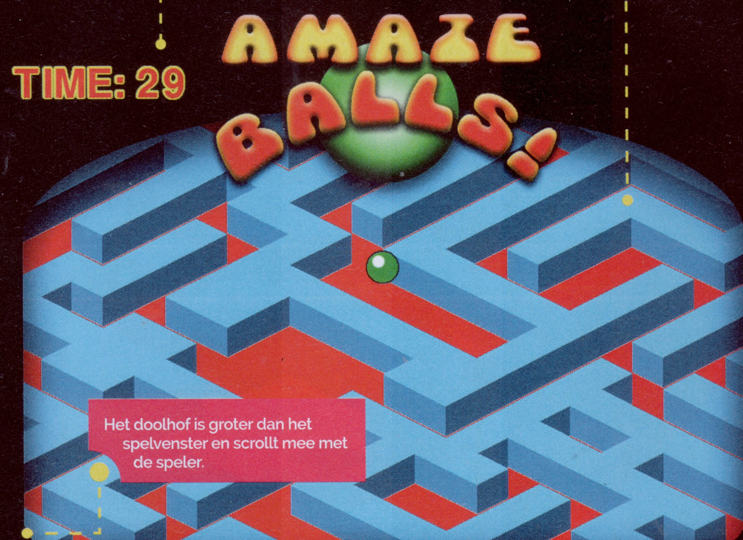
```python
import json
import os

def loadmap(mp):
    with open(mp) as json_data:
        d = json.load(json_data)
    mapdata = {"width":d["width"], "height":d["height"]}
    rawdata = d["layers"][0]["data"]
    mapdata["data"] = []
    for x in range(0, mapdata["width"]):
        st = x*mapdata["width"]
        mapdata["data"].append(rawdata[st:st+mapdata["height"]])

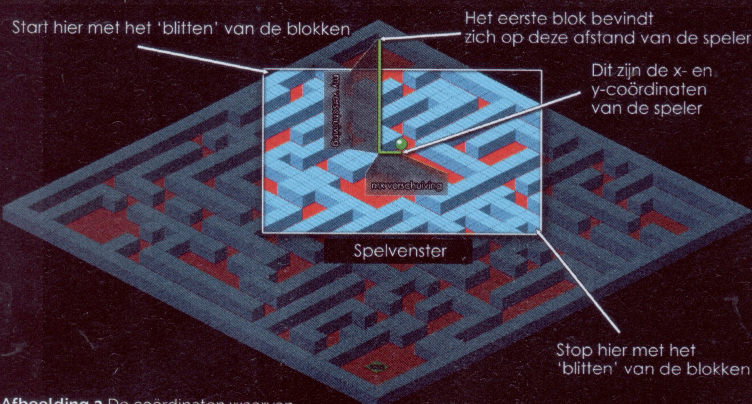
    tileset = "maps/" + d["tilesets"][0]["source"].replace(".tsx",".json")
    with open(tileset) as json_data:
        t = json.load(json_data)

    mapdata["tiles"] = t["tiles"]
    for tile in range(0,len(mapdata["tiles"])):
        path = mapdata["tiles"][tile]["image"]
        mapdata["tiles"][tile]["image"] = os.path.basename(path)
        mapdata["tiles"][tile]["id"] = mapdata["tiles"][tile]["id"]+1
    return mapdata
```

▲ De functie `loadmap()` vertaalt de data van Tiled naar het dataformaat voor onze kaart.







Afbeelding 2 De coördinaten waarvan de kaart start worden berekend als een verschuiving ten opzichte van de speler, en de blokken van het doolhof worden alleen binnen de rechthoek getekend die in het spelvenster zichtbaar is.

we kunnen het in zijn geheel inladen en gewoon de delen gebruiken die we nodig hebben. Laten we een nieuwe module maken om kaarten af te handelen, zoals we in het Pygame Zero-artikel over Pac-Man (zie MagPi 7) hebben gedaan: de module **map3d.py**. Python heeft een ingebouwde module om json-bestanden te importeren, dus we kunnen **import json** bovenaan het bestand **map3d.py** typen om de module in te laden. We hebben ook de module os nodig om met bestandsnamen te werken, dus die importeren we ook. Dan dienen we alleen nog maar een functie te schrijven om onze kaart in te laden.

Tip

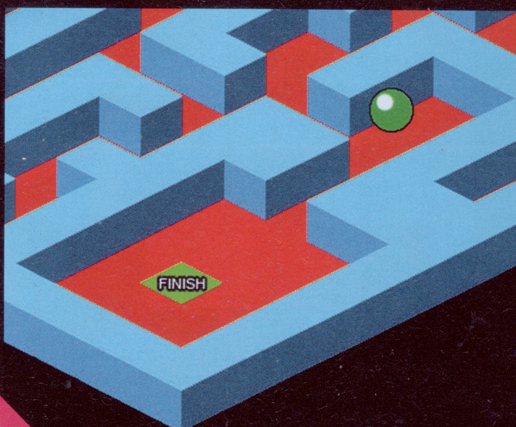
Bekijk de json

Je kunt json-bestanden in elke teksteditor bekijken, maar een programmeereditor werkt waarschijnlijk het best. Probeer Geany eens.

07 Krijgen wat we nodig hebben

Laten we een functie `loadmap()` maken en een parameter `mp` gebruiken om de locatie van een bestand aan de functie door te geven. Kijk eens naar de code in **figure1.py** om te zien hoe we dat schrijven. Je ziet daar dat we onze kaartdata in een variabele `d` inladen met de functie `json.load()`. Daarna kunnen we de breedte en hoogte in een dictionary mapdata kopiëren. Aan het einde van de functie bevat deze dictionary alle gegevens die we nodig hebben. Omdat we een tijdelijke kopie van de blokgegevens gemaakt hebben (`rawdata`), kunnen we daarna door de waardes gaan en ze in het formaat omzetten dat we in `mapdata` nodig hebben.

We hebben een nieuwe tegel voor de finish toegevoegd. Als de speler zich op dit blok beweegt, heeft hij het doolhof opgelost.



08 Een verzameling tegels

De eerste informatie die we nodig hebben, is waar we de details kunnen vinden over welke afbeelding we voor elk blok dienen te gebruiken. Dat vind je in de waarde `tilesets`. In dit geval veronderstellen we dat we slechts één verzameling tegels gedefinieerd hebben. We kunnen die dan eenvoudigweg inlezen en onze blokafbeeldingen vinden. We lopen hier wel tegen een klein probleempje aan: onze kaartgegevens verwijzen naar het tegelbestand als een tsx-bestand. Open dus Tiled opnieuw en exporteer de verzameling tegels als een json-bestand. Wanneer we het dan importeren, vervangen we de tsx-extensie eenvoudigweg door json.

09 Een nacht op de tegels

Zodra de tegels als een json-bestand ingeladen zijn, kunnen we erdoor gaan en de namen van elk van onze blokafbeeldingen opvragen. Merk op dat we 1 optellen bij de waarde `id` om overeen te komen met de waardes die Tiled geëxporteerd heeft. Wanneer we al die gegevens in de dictionary `mapdata` hebben, kunnen we die teruggeven aan ons hoofdprogramma met de opdracht `return mapdata`. In het hoofdprogramma moeten we de module map3d nog bovenaan de code importeren en dan kunnen we vóór de functie `draw()` gewoon `mapData = map3d.loadmap("maps/map1.json")` schrijven in plaats van onze lijst van enen en nullen.

10 Groot denken

In deel één van deze reeks was ons doolhof 12 bij 12 blokken groot, wat perfect in het speelveld paste. We hebben nu een doolhof van 30 bij 30 dat, als we het volledig tekenen, buiten het speelveld komt. We moeten dus de kaart rond het scherm kunnen scrollen terwijl de speler door het doolhof beweegt. We kunnen dat doen door de stuiterbal in het centrum van het speelveld te houden en — terwijl de speler beweegt — de kaart te scrollen. Wat we dus eigenlijk zeggen, is dat we de kaart relatief ten opzichte van de speler gaan tekenen in plaats van relatief ten opzichte van het spelvenster.

11 Het is allemaal relatief

Om onze kaart te tekenen, gebruiken we dezelfde fundamentele lussen (volgens x en y) als tevoren, maar we starten met onze kaart te tekenen gebaseerd op de coördinaten die we voor de x- en y-schermcoördinaten van de speler berekenden. Met die startpositie op het scherm overlopen we in de lus een bereik aan beide kanten van de speler, in beide richtingen. Zie **afbeelding 2** op de vorige pagina voor een visuele uitleg van wat we in deze lussen doen. In eenvoudige termen zeggen we hier: "Start met de kaart te tekenen op de coördinaten die de speler in het midden van het venster doen verschijnen. Teken dan alleen de blokken die in het venster zichtbaar zijn." Zie **figure3.py** voor hoe we de functie `drawMap()` aangepast hebben om dat te doen.

12 Extra functies

In **figure3.py** zie je dat we enkele nieuwe functies gebruiken die we nog niet gedefinieerd hebben. De eerste is `onMap()`, waar we een x- en y-coördinaat aan doorgeven. Dit zijn bloklocaties die we testen om zeker te zijn dat de coördinaten waarnaar we vragen zich wel op onze kaart bevinden; anders krijgen we een foutmelding. Als de x of y kleiner dan 0 zijn of groter dan de breedte respectievelijk hoogte van de kaart, dan kunnen we het blok negeren. De andere nieuwe functie is `findData()`. Die functie vindt de data geassocieerd met een tegel met een gegeven id. Kijk in **figure4.py** (op de volgende pagina) hoe deze functies geschreven zijn.

13 Transparantie

Als we nu onze kaart tekenen, zijn we die mooie diamantvorm van onze kaart kwijt. En als we de speler naar onderen van de kaart bewegen, krijgen we bovenaan een gekartelde rand en komen blokken plots in en uit het gezichtsveld terwijl de functie `drawMap()` beslist welke hij tekent. We kunnen dit wat oppoetsen door er een frame over te tekenen dat de randen van het gebied verduistert. Dat doen we met een afbeelding die het hele venster overlapt maar die een transparant deel uitgeknipt heeft in het gebied waar we de kaartblokken willen tonen. We 'blitten' dit frame nadat we `drawMap()` in onze functie `draw()` aangeroepen hebben.

14 Ik kan niet bewegen!

Als je na het tekenen van de kaart de code uit deel 1 toevoegt, merk je misschien dat je de stuiterbal niet meer kunt bewegen. Dat komt

figure3.py

```
def drawMap():
    psx = OFFSETX
    psy = OFFSETY-32
    mx = psx - player["sx"]
    my = psy - player["sy"]+32

    for x in range(player["x"]-12, player["x"]+16):
        for y in range(player["y"]-12, player["y"]+16):
            if onMap(x,y):
                b = mapData["data"][y][x]
                td = findData(mapData["tiles"], "id", b)
                block = td["image"]
                bheight =  td["imageheight"]-34
                bx = (x*32)-(y*32) + mx
                by = (y*16)+(x*16) + my
                if -32 <= bx < 800 and 100 <= by < 620:
                    screen.blit(block, (bx, by - bheight))
                if x == player["x"] and y == player["y"]:
                    screen.blit("ball"+str(player["frame"]),
    (psx, psy))
```

▲ Onze nieuwe functie `drawMap()`

map3d.py

> Taal: **Python**

```
# 3dmap module for AmazeBalls
import json
import os

def loadmap(mp):
    with open(mp) as json_data:
        d = json.load(json_data)
    mapdata = {"width":d["width"], "height":d["height"]}
    rawdata = d["layers"][0]["data"]
    mapdata["data"] = []
    for x in range(0, mapdata["width"]):
        st = x*mapdata["width"]
        mapdata["data"].append(rawdata[st:st+mapdata["height"]])

    tileset = "maps/" + d["tilesets"][0]["source"].replace(
  ".tsx",".json")
    with open(tileset) as json_data:
        t = json.load(json_data)

    mapdata["tiles"] = t["tiles"]
    for tile in range(0,len(mapdata["tiles"])):
        path = mapdata["tiles"][tile]["image"]
        mapdata["tiles"][tile]["image"] = os.path.basename(path)
        mapdata["tiles"][tile]["id"] = mapdata["tiles"][tile]["id"]+1
    return mapdata
```



figure4.py

```python
def onMap(x,y):
    if 0 <= x < mapData["width"] and 0 <= y < mapData["height"]:
        return True
    return False

def findData(lst, key, value):
    for i, dic in enumerate(lst):
        if dic[key] == value:
            return dic
    return -1
```

De functies om te testen of coördinaten zich in het kaartgebied bevinden — **onMap()** — en om tegelgevens te vinden om de kaart te tekenen — **findData()**.

figure5.py

```python
def doMove(p, x, y):
    if onMap(p["x"]+x, p["y"]+y):
        mt = mapData["data"][p["y"]+y][p["x"]+x]
        if mt == 1:
            p.update({"queueX":x, "queueY":y, "moveDone":False})
            if mt == 3:
                mazeSolved = True
```

De nieuwe functie **doMove()**.

omdat de data die we ingeladen hebben een lichtjes ander formaat gebruiken en vloerblokken met id 1 en muren met id 2 heeft. Daarom denkt onze functie `doMove()` momenteel dat we door muren omsingeld zijn (die we in deel 1 voorgesteld hadden door id 1). Verander dus de functie `doMove()` om met het nieuwe dataformaat om te gaan. Kijk in **figure5.py** wat we dienen te schrijven.

15 Vind de uitgang

Nu onze bal weer beweegt, dienen we nog enkele standaardwaardes aan te passen waarmee we het spel starten. In het vorige deel lieten we de speler starten op x = 0 en y = 3. Om met onze nieuwe kaart te werken, dien je die waardes naar 3 en 3 te veranderen in de spelergegevens bovenaan de code. Verander ook de constante `OFFSETY` naar 300 om de kaart een beetje meer naar beneden te bewegen. Je zou nu de stuiterbal door het doolhof moeten kunnen leiden naar de onderkant van het scherm, waar we de finish vinden.

16 Over de finishlijn

Als we nu op het eindblok proberen te bewegen, zijn we daar niet toe in staat. Onze functie `doMove()` denkt immers dat we alleen maar naar blokken met id 1 kunnen bewegen. We dienen dus nog een andere voorwaarde toe te voegen. In plaats van alleen te testen of mt gelijk aan 1 is, testen we nu op `mt == 1 or mt == 3` (omdat het id van het eindblok 3 is). We kunnen dan een variabele een waarde geven wanneer de speler op het eindblok stapt: `if mt == 3: mazeSolved = True`. We dienen `mazeSolved` ook als een globale variabele in te stellen in `doMove()` en zijn initiële waarde bovenaan ons programma op False te zetten.

17 De tijd raakt op

Laten we nu nog een timer aan ons spel toevoegen. Wanneer de speler de finish bereikt (`mazeSolved is True`), kunnen we de timer stoppen en een boodschap tonen. Creëer dus eerst bovenaan ons programma een timervariabele met `timer = 0`. Helemaal op het einde, vlak voor `pgzrun.go()`, kunnen we de klokfunctie van Pygame Zero gebruiken, `schedule_interval()`. Als we `clock.schedule_interval(timerTick, 1.0)` schrijven, wordt de functie `timerTick()` elke seconde aangeroepen.

18 Om het snelst

Dan rest ons nog één taak: de functie `timerTick()` definiëren. We dienen daarin te controleren of `mazeSolved` gelijk aan `False` is en 1 bij onze variabele `timer` op te tellen als dat zo is. Dan kunnen we een regel `screen.draw.text` aan onze functie `draw()` toe te voegen om de timerwaarde te tonen. En als `mazeSolved` gelijk aan True is, kunnen we nog meer tekst toevoegen om te melden dat het doolhof opgelost is en hoeveel seconden je nodig had. Zie de volledige programmacode voor hoe je de code daarvoor nog schrijft.

In het volgende nummer van MagPi gaan we wat slechteriken toevoegen om tegen te spelen. En voor de fun gooien we er ook nog wat dynamiet bij!

Tip

Volgorde van tekenen

Vergeet niet dat in de functie **draw()** de dingen in de volgorde worden getekend waarin je ze oproept. Teken dus altijd de dingen die je bovenaan wilt zien als laatste.

amazeballs2.py

> Taal: **Python**

DOWNLOAD DE VOLLEDIGE CODE:



magpi.cc/fPBrhM

```
import pgzrun
import map3d

player = {"x":3, "y":3, "frame":0, "sx":0, "sy":96,
          "moveX":0, "moveY":0, "queueX":0, "queueY":0,
          "moveDone":True, "movingNow":False, "animCounter":0}
OFFSETX = 368
OFFSETY = 300
timer = 0
mazeSolved = False

mapData = map3d.loadmap("maps/map1.json")

def draw(): # Pygame Zero draw function
    screen.fill((0, 0, 0))
    drawMap()
    screen.blit(‘title’, (0, 0))
    screen.draw.text("TIME: "+str(timer) , topleft=(20, 80), owidth=0.5, ocolor=(255,255,0), color=(255,0,0) , fontsize=60)
    if mazeSolved:
        screen.draw.text("MAZE SOLVED in " + str(timer) + " seconds!" , center=(400, 450), owidth=0.5, ocolor=(0,0,0), color=(0,255,0) , fontsize=60)


def update(): # Pygame Zero update function
    global player, timer
    if player["moveDone"] == True:
        if keyboard.left: doMove(player, -1, 0)
        if keyboard.right: doMove(player, 1, 0)
        if keyboard.up: doMove(player, 0, -1)
        if keyboard.down: doMove(player, 0, 1)
    updateBall(player)

def timerTick():
    global timer
    if not mazeSolved:
        timer += 1

def drawMap():
    psx = OFFSETX
    psy = OFFSETY-32
    mx = psx - player["sx"]
    my = psy - player["sy"]+32

    for x in range(player["x"]-12, player["x"]+16):
        for y in range(player["y"]-12, player["y"]+16):
            if onMap(x,y):
                b = mapData["data"][y][x]
                td = findData(mapData["tiles"], "id", b)
                block = td["image"]
                bheight = td["imageheight"]-34
                bx = (x*32)-(y*32) + mx
                by = (y*16)+(x*16) + my
                if -32 <= bx < 800 and 100 <= by < 620:
                    screen.blit(block, (bx, by - bheight))
                if x == player["x"] and y == player["y"]:
                    screen.blit("ball"+str(player["frame"]), (psx, psy))

def findData(lst, key, value):
    for i, dic in enumerate(lst):
        if dic[key] == value:
            return dic
    return -1

def onMap(x,y):
    if 0 <= x < mapData["width"] and 0 <= y < mapData["height"]:
        return True
    return False

def doMove(p, x, y):
    global mazeSolved
    if onMap(p["x"]+x, p["y"]+y):
        mt = mapData["data"][p["y"]+y][p["x"]+x]
        if mt == 1 or mt == 3:
            p.update({"queueX":x, "queueY":y, "moveDone":False})
            if mt == 3:
                mazeSolved = True

def updateBall(p):
    if p["movingNow"]:
        if p["moveX"] == -1: moveP(p,-1,-0.5)
        if p["moveX"] == 1: moveP(p,1,0.5)
        if p["moveY"] == -1: moveP(p,1,-0.5)
        if p["moveY"] == 1: moveP(p,-1,0.5)
    p["animCounter"] += 1
    if p["animCounter"] == 4:
        p["animCounter"] = 0
        p["frame"] += 1
        if p["frame"] > 7:
            p["frame"] = 0
        if p["frame"] == 4:
            if p["moveDone"] == False:
                if p["queueX"] != 0 or p["queueY"] !=0:
                    p.update({"moveX":p["queueX"], "moveY":p["queueY"], "queueX":0, "queueY":0, "movingNow": True})
            else:
                p.update({"moveDone":True, "moveX":0, "moveY":0, "movingNow":False})

        if p["frame"] == 7 and p["moveDone"] == False and p["movingNow"] == True:
            p["x"] += p["moveX"]
            p["y"] += p["moveY"]
            p["moveDone"] = True

def moveP(p,x,y):
    p["sx"] += x
    p["sy"] += y

clock.schedule_interval(timerTick, 1.0)
pgzrun.go()
```

Vintagetelevisie

Martin Mander gebruikt in zijn nieuwste project de nieuwe TV HAT voor een huwelijk tussen analoge aansturing en digitale televisie. **David Crookes** stemt af.



MAKER

Martin Mander

Martin Mander werkt als analist in Noorwegen. Hij houdt ervan om vintagetechnologie nieuw leven in te blazen door ze te combineren met nieuwe apparaten.

magpi.cc/fetqPS

Toen de draagbare Hitachi-televisie van Martin Mander in 1975 geproduceerd werd, had je in Nederland maar twee en in Vlaanderen maar één televisiekanaal. En als je van kanaal wilde veranderen, diende je het comfort van je bank te verlaten.

Vandaag hebben we de keuze uit talloze televisiekanalen en zelfs een coole Raspberry Pi TV HAT (**www.elektor.nl/rpi-tv-hat**) die ons via een geschikte antenne laat genieten van DVB-T2-uitzendingen. Dus wat besloot de nostalgische Martin om te doen toen hij zijn nieuw aangekochte TV HAT met de 40-pens gpio-header van de Pi verbond? Hij stak het in zijn oude tv-toestel met een stevige draaiknop en beperkte het aantal kanalen tot degene die hij op één hand kon tellen — "de 1982-ervaring" noemde hij het, omdat hij naar het in dat jaar gelanceerde Channel 4 wilde kijken.

Live tv

Martin is een crack in het ombouwen van crt-televisies — hij heeft er sinds 2012 al zes gemaakt, die hij inbouwde in monitors, fotolijstjes en neonreclamepanelen. "Voor mijn nieuwste project wilde ik iets leuks doen met de nieuwe HAT en zien of ik de tv-streams eenvoudig kon tonen en aansturen op sommige van mijn omgebouwde televisies", zegt hij. Het resultaat is nu gepromoveerd naar zijn kantoor, om op de achtergrond wat tv te kijken terwijl hij werkt. "Het was heel leuk om de TV HAT-streams met de draaiknop te laten werken", voegt hij toe. Hoewel Martin zich onmiddellijk op de HAT stortte zonder de instructies te lezen of een antenne aan te sluiten, volgde hij uiteindelijk de handleiding en ontdekte hij dat het vrij eenvoudig was om het aan de gang te krijgen. Toen besloot hij om zijn Hitachi Pi-project (**magpi.cc/vVobwP**), dat hij al van een 8 inch 4:3-scherm voorzien had, een nieuwe bestemming te geven.

"Dat project is gebaseerd op een Pi 3 en heeft al een draaiknop die met de gpio verbonden is", legt hij uit. "Dat betekende dat ik met de TV HAT kon experimenteren, maar indien nodig altijd kon terugvallen op het script van het originele project, zonder dat ik aan de hardware veranderingen diende aan te brengen."

Van kanaal veranderen

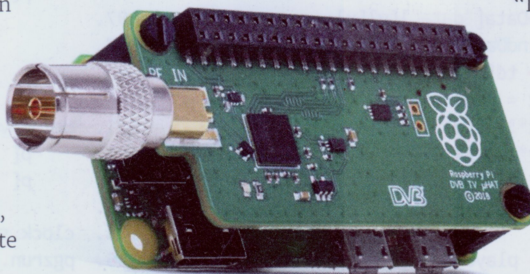
Martins belangrijkste taak was inderdaad ervoor zorgen dat hij met de draaiknop van kanaal kon veranderen. Dat bleek eenvoudiger te bereiken dan hij verwachtte. "Als je een tv-programma via de webinterface van Tvheadend bekijkt, downloadt het een m3u-afspeellijst dat je dan in VLC of elke andere mediaspeler kunt bekijken", zegt hij.

> Het was heel leuk om de TV HAT-streams met de draaiknop te laten werken

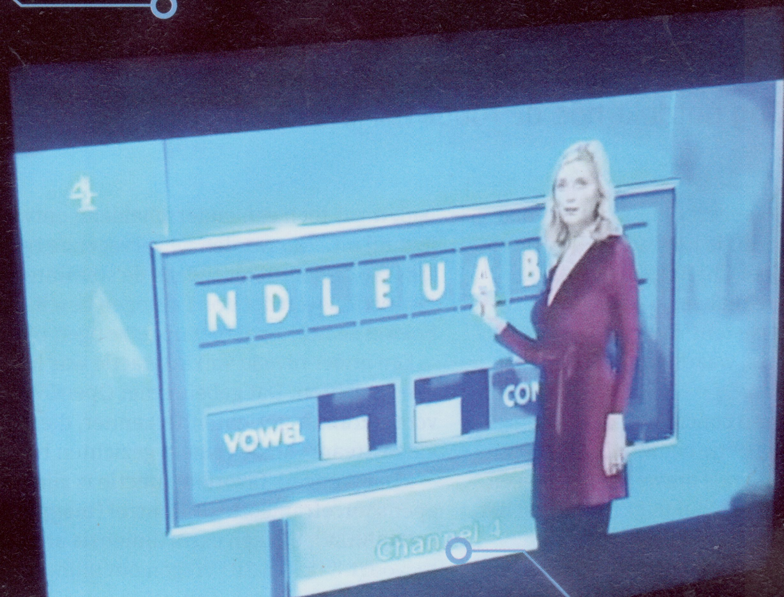
"In het begin dacht ik dat het bestand met de afspeellijst specifiek voor het individuele tv-programma was, omdat de naam van het programma in het bestand opgenomen was. Maar eigenlijk is elk afspeellijstbestand specifiek voor het kanaal zelf. Ik kon dus voor elk kanaal een afspeellijst downloaden en ze in een map opslaan om me alle kijkopties te geven."

Hij sloeg dus de afspeellijsten op van de vier belangrijkste Britse kanalen van 1982 (BBC1, BBC2, ITV en Channel 4) en hernoemde ze naar channel1, channel2, channel3 en channel4.

"Daarna creëerde ik een script met een oneindige lus die uitkijkt naar actie op de gpio-pin die met de draaiknop verbonden is", gaat hij door. "Als het script detecteert dat de schakelaar bewogen is, opent dit



Het project was mogelijk dankzij de nieuwe Raspberry Pi TV HAT.



De Hitachi-televisie is van een Pimoroni 8 inch 4:3-scherm voorzien en een Raspberry Pi 3.

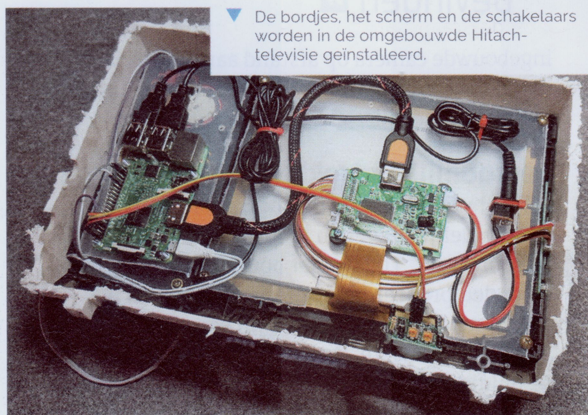
De programma's worden gestreamd door een Pi 2-server. De kanalen verander je met de draaiknop.

De naam van het kanaal verschijnt kort onderaan het scherm — de bestanden met de afspeellijsten zijn bewerkt in Notepad.

▲ Een jonge Marin Mander besluit dat het lege scherm van zijn zwart-witte Philips TX met zes manueel ingestelde knoppen interessanter is dan de tv-programma's. In de toekomst wil hij een van deze toestellen ombouwen.

het eerste afspeellijstbestand in VLC op volledige schermgrootte. De volgende keer dat de schakelaar beweegt, voegt het script 1 toe aan het laatste cijfer in de bestandsnaam, zodat dit het volgende kanaal in de map opent."

Martin is nu al de volgende fase van het project aan het plannen. Hij overweegt om het script om van kanaal te veranderen uit te breiden zodat het ook toegang geeft tot streams van zijn ip-camera's. Daarnaast wil hij de momenteel herlaadbare luidspreker vervangen door een Speaker pHAT en wil hij ook de originele volumedraaiknop met de audio van de Pi laten werken. "Het gaf me echt voldoening om dit project werkend te krijgen, en ik zie nog heel wat mogelijkheden voor me", besluit hij.



▼ De bordjes, het scherm en de schakelaars worden in de omgebouwde Hitach-televisie geïnstalleerd.

In een **oogopslag**

- Een met ethernet verbonden Pi 2 functioneert als tv-server.
- Die stuurt de streams naar een retrotelevisie met ingebouwde Pi.
- Met een draaiknop verander je van kanaal op het 8 inch-scherm.
- Een Python-script handelt het van kanaal veranderen af.
- Zowel standaard- als hd-streams zijn mogelijk.

R3-14, een persoonlijke robotassistant

Deze persoonlijke robotassistent heeft niet alleen een vriendelijk gezicht, maar ook talloze nuttige functies. **Phil King** ontdekt hoe de robot tot leven kwam.



MAKER

Sanjeet Chatterjee

Een scholier met een passie voor technologie en ontwerp. In zijn vrije tijd houdt hij van programmeren, met elektronica experimenteren en viool spelen.

magpi.cc/BQLMaD

R3-14 staat voor 'Raspberry Pi' (maar dat had je al wel door?) en dit is het eerste Pi-project van scholier Sanjeet Chatterjee. Op zoek naar iets origineels om te maken, koos hij voor een persoonlijke robotassistent met een fysieke aanwezigheid in de kamer, waarbij interactie een belangrijke eigenschap is.

"In het begin had ik geen concreet plan", zegt Sanjeet, "maar beetje bij beetje bouwde ik functies in, terwijl ik over verschillende ontwerpen en ideeën nadacht. Toen ik eindelijk een duidelijke visie van het finale product had, zette ik alles in elkaar."

Hij heeft er zeker een heleboel functies ingebouwd. R3-14 werkt niet alleen als slimme luidspreker, met Google Assistant om te reageren op gesproken vragen, maar heeft ook een indrukwekkend gamma aan mogelijkheden. Met Siri-stemopdrachten kun je de verlichting en apparaten thuis via 433 MHz-transceivers in- en uitschakelen. Via een webapp kun je het gemotoriseerde hoofd van de robot en zijn ingebouwde camera op afstand aansturen. En het hoofd volgt automatisch het gezicht van de gebruiker terwijl het met allerlei berichten antwoordt.

> "Robotassistenten kunnen zich op een dag in het middelpunt van ons leven bevinden"

Persoonlijk betrokken

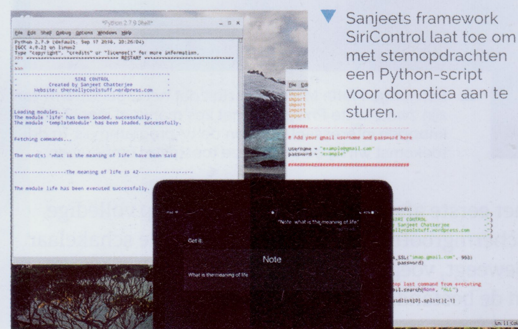
Sanjeet vindt het heel belangrijk dat de robot er vriendelijk uitziet en in interactie gaat met de gebruikers. Tijdens het bouwproces van vier maanden probeerde hij allerlei ontwerpen uit, die hij schetste en waarvoor hij een prototype uit aluminium maakte. Zo kwam hij uiteindelijk tot de huidige versie, die gebouwd is met 3d-geprinte onderdelen uit PLA.

"Het was belangrijk dat de robot menselijke trekjes had, maar toch een onderscheidend ontwerp", legt hij uit. "Zo worden de vervagende rgb-leds die de gebruiker tijdens de steminteractie feedback geven door ons als ogen beschouwd, hoewel ze helemaal niet op onze ogen lijken."

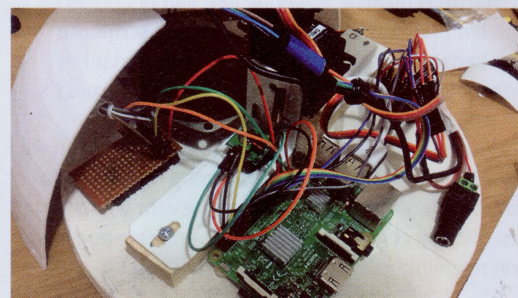
Een ander niveau van interactie was het volgen van gezichten. Dat gebeurt met OpenCV en de voorgetrainde gezichtenherkenner, die in de live videostream van de camera gezichten herkent.

Maar Sanjeets favoriete functie is zijn eigen opensourceframeork SiriControl (**magpi.cc/2t3Bh4v**), dat een extra dimensie aan domotica toevoegt. "Ik kan Siri vragen om mijn apparaten in huis aan te sturen, van gelijk waar in de wereld!"

SiriControl werkt als volgt: het vraagt Siri om een notitie aan te maken, die met een Gmail-account gesynchroniseerd wordt en dan met Python via het IMAP-protocol wordt opgevraagd.



Sanjeets framework SiriControl laat toe om met stemopdrachten een Python-script voor domotica aan te sturen.



De basis bevat een Pi 3, luidspreker en een boel draden.

R3-14's hoofd bevat een camera om live video te streamen en gezichten te herkennen.

Twee servo's met een pan-tiltmechanisme bewegen het hoofd.

Het lichaam en het hoofd van de robot zijn met 3d-geprinte onderdelen uit PLA gemaakt.

In een oogopslag

- Sanjeet werkte aan dit project in zijn schoolvakanties.
- Hij had geen voorgaande ervaring met elektronica.
- Twee servo's draaien en kantelen R3-14's hoofd.
- Hij schreef een Python-bibliotheek om de rgb-ledkleuren te laten vervagen.
- Zijn volgende project is een slimme hoofdband om IoT-apparaten aan te sturen.

▼ Sanjeet test het aluminium prototype met alle componenten.

De juiste module wordt dan uitgevoerd, afhankelijk van de woorden die je uitgesproken hebt. Dat betekent dan in de praktijk dat R3-14 RF-signalen naar de juiste slimme stekker stuurt. "Ik kwam op deze oplossing toen ik met de imap-module aan het experimenteren was", herinnert Sanjeet zich. "Daarna ontwikkelde ik een bibliotheek zodat anderen ook eenvoudig Siri-stemopdrachten aan gelijk welk project kunnen toevoegen."

Onafhankelijk wonen

Hoewel R3-14 een leuk en boeiend karakter is om mee te spreken, gelooft Sanjeet dat gelijkaardige robots ook voor serieuzer gebruik in huis geschikt zijn.

"Robotassistenten kunnen zich op een dag in het middelpunt van ons leven bevinden", zegt hij, "en ik ben er zeker van dat ze nuttig zijn om realtime problemen te ontdekken en in het algemeen heel behulpzaam te worden, bijna als een persoonlijke assistent. Robots zoals R3-14 kunnen na wat extra ontwikkeling ook gezelschap geven aan ouderen, die daardoor langer onafhankelijk en alleen kunnen blijven wonen."

Raspberry Pitaar

Wat kun je aanvangen met een Pi Zero en een oude elektrische gitaar? **Nicola King** kanaliseert haar innerlijke Dave Grohl om het te ontdekken.



MAKER

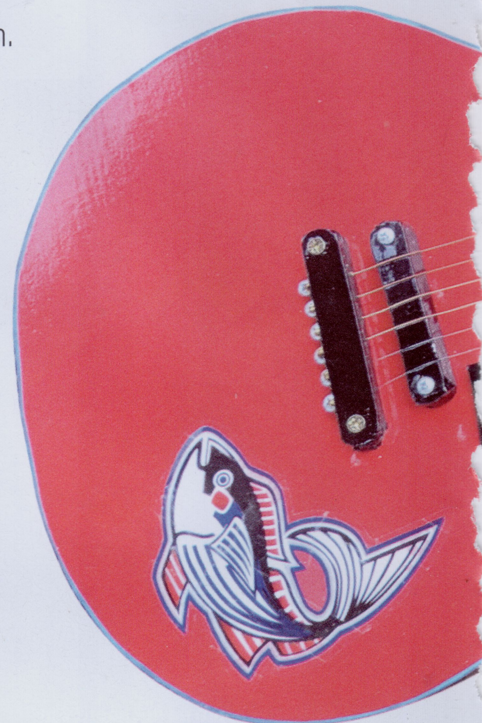
Behruz Farshi

Software-ontwikkelaar Behruz woont in Oostenrijk en houdt ervan om in zijn vrije tijd te knutselen, games te ontwikkelen, muziek te spelen en fit te blijven.

magpi.cc/fokvqn

Toen Behruz Farshi de oude elektrische gitaar van zijn broer oppakte, begon hij na te denken over de eindeloze mogelijkheden als hij er een digitaal instrument van zou kunnen maken. Bij zijn eerste poging om gitaar te spelen vond hij het zwaar werk voor zijn vingers: "Ik dacht dat het leuk zou zijn om een soort instrument te hebben dat je losjes kunt spelen, maar dat nog steeds het echte werk is", vertelt hij ons. "Een digitale gitaar zou niet zoveel spanning in de snaren nodig hebben en zou ook gelijk welke geluiden kunnen maken die je erin programmeert! Daarom bestelde ik een Pi Zero en ontwikkelde een prototype."

Het was Behruz' eerste project met een Raspberry Pi, legt hij uit: "Ik koos voor de Pi Zero omdat die klein is, alle hardware heeft om snel iets te maken en Linux draait."

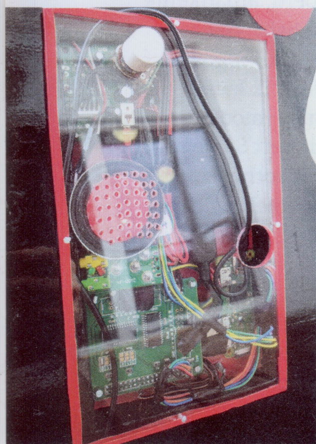


> Omdat het idee was om onderdelen van een echte gitaar te gebruiken en de ruimte heel beperkt is, was het niet eenvoudig om te monteren

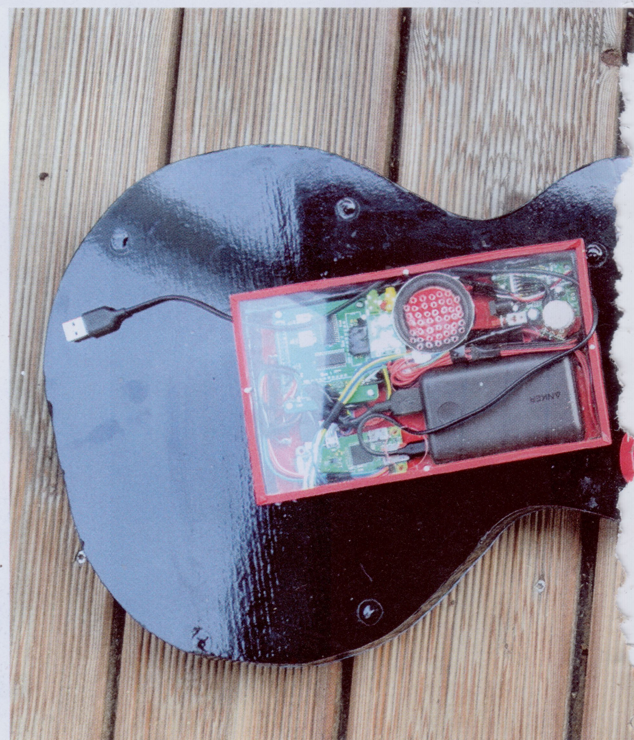
Gewapend met zijn nieuwe Pi Zero had Behruz vier dagen nodig om zowel de hardware als de software voor zijn project te ontwikkelen. Het meest tijdrovende aspect was het hardware-element van het fretboard: "Omdat het idee was om onderdelen van een echte gitaar te gebruiken en de ruimte heel beperkt is, was het niet eenvoudig om te monteren."

Muzikale matrix

De Pitaar heeft een toetsenblokmatrix van 22 frets en 6 snaren. Wanneer de speler een snaar op een fret duwt, zijn ze verbonden, wat betekent dat er stroom tussen kan vloeien. De Raspberry Pi detecteert dat op een van de gpio-pennen en speelt dan de bijbehorende noot. "Om uit te vinden welke frets ingedrukt zijn, doorlopen we één voor één alle snaren, zetten we de bijbehorende gpio hoog en controleren we dan welke fret een hoge waarde op zijn gpio krijgt", legt Behruz uit. "De rest is een eenvoudig softwareproject om geluiden te maken voor de ingedrukte frets."



▲ De Pi Zero is in een nis aan de achterkant van de gitaar gemonteerd en speelt digitale noten door een miniluidspreker.



► Aan de achterkant van de voltooide Pitaar zie je de Pi Zero, afgedekt met een stukje plexiglas.

In een oogopslag

- De Pitaar kan midi-geluiden afspelen, inclusief een piano.
- Voorlopig is het instrument eenstemmig, maar een meerstemmige versie is mogelijk.
- De matrix heeft meer gpio-pennen nodig dan de Pi Zero heeft…
- Daarom gebruikte Behruz een IO Pi Plus-bordje om het aantal pennen uit te breiden.
- De stroom komt van een batterijpakket met een schakelaar.

Om de matrix te creëren, boorde hij gaten in het fretboard en soldeerde hij van onderen draden aan de frets. Nadat hij het eerste prototype gebouwd had, ontdekte hij dat de frets soms de snaren verbonden en de gitaar daardoor deden uitvallen. "De oplossing is heel eenvoudig: snij elke fret in zes onderdelen die gescheiden zijn en verbind ze in de plaats met diodes, zodat ze de stroom maar in één richting geleiden, en scheid de snaren. Als je dat manueel met echte fretdraad doet, duurt dat wel eeuwen."


Deze standaard elektrische gitaar is aangepast tot een digitaal instrument.

Toegift

Met deze ervaring plant Behruz nu een tweede versie van de Pitaar. "Het lichaam van de gitaar moet dan zo dicht mogelijk bij dat van een echte elektrische gitaar komen — ik ben er bijna. En het fretboard met de gescheiden frets uiteraard — zonder die aanpassing is het niet mogelijk om de frets dicht bij elkaar op een betrouwbare manier te spelen, omdat de snaren dan onbedoeld met elkaar verbinden. Aan de kant van de software is mijn plan om van het hele project een midi-gitaar te maken, zodat ik het eenvoudigweg over usb met een pc of smartphone kan verbinden en gelijk welke geluiden kan afspelen."

Zelfs zoals hij nu is, klinkt deze ingenieuze upgrade van een oude elektrische gitaar ons al als muziek in de oren.

Tortelduifjes

Tekstberichtjes zijn niet altijd het beste communicatiemiddel voor een koppel. **Koen Vervloesem** ontdekt hoe een doosje met tortelduifjes de vrede in huis kan bewaren.



MAKER

Olivier Ros

Olivier is elektronica-ingenieur van opleiding en kunstenaar en ondernemer in hart en nieren. De Fransman gaf drie jaar geleden zijn saaie 9-to-5-job op om voltijds uitvinder te worden. Hij verkoopt online wat hij maakt, vaak gebaseerd op de Raspberry Pi.

roscop13 op Instagram

Toen de vriendin van Olivier Ros in Rio de Janeiro leefde terwijl hij in Parijs woonde, was communiceren moeilijk omdat ze niet altijd op hetzelfde moment beschikbaar waren. "Uiteraard konden we onze telefoon gebruiken, maar dat is niet zo romantisch: tekstberichten die je gehaast intypt, zijn gemakkelijk verkeerd te interpreteren. Daarom vond ik Love Birds uit, waardoor we met onze stem konden communiceren: door de intonatie die je stem in een spraakbericht legt, vermijd je heel wat misverstanden. De spraakberichten worden bovendien opgeslagen zoals bij een antwoordapparaat."

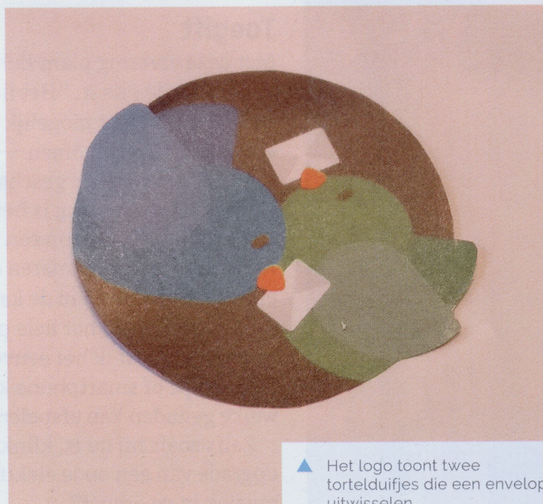
Olivier kwam op het idee toen hij aan die ouderwetse antwoordapparaten met cassette terugdacht: "Hoe leuk was het niet om thuis te komen en dan op die knop te drukken om te ontdekken dat het grote apparaat nieuwe berichten bevatte! Ik hield ook van het idee van die rode telefoon in films met een rechtstreekse beveiligde lijn tussen landen om diplomatieke incidenten te vermijden. Love Birds is dus een mix van dit alles met als doel om diplomatieke incidenten bij koppels te vermijden :)"

Klein genoeg voor in een doosje

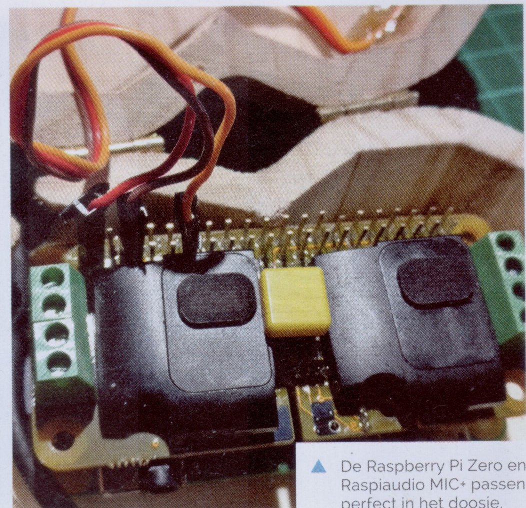
De Raspberry Pi was een natuurlijke keuze voor Olivier, omdat het apparaat volledige ondersteuning voor I²S-audio heeft (zowel invoer als uitvoer) en omdat er bibliotheken voor Telegram voor bestaan (Olivier gebruikte Telethon). "Je zou dit project eenvoudig naar elke Linux-computer kunnen porten. Een Arduino daarentegen is niet krachtig genoeg voor deze taak. Met een ESP32 zou het mogelijk zijn, maar het programmeren zou meer tijd vragen."

> "Olivier kwam op het idee van Love Birds toen hij aan die ouderwetse antwoordapparaten met cassette terugdacht"

Van alle modellen is de Raspberry Pi Zero de perfecte kandidaat, omdat die goedkoop is en klein genoeg om in een doosje te passen. "Ik gebruikte de Raspiaudio MIC+ voor de audio-uitgang, de ingebouwde microfoon en de 5 W-luidsprekers, allemaal in hetzelfde formaat als de Pi Zero." Olivier voegde ook een optionele 5 V-servomotor toe die rechtstreeks door de 5 V-voeding en een gpio-pen van de Raspberry Pi aangedreven wordt. "Wanneer je een bericht ontvangt, begint het logo van de twee tortelduifjes te draaien. Het geluid van de motor is luid genoeg om je te waarschuwen dat je een bericht hebt."



▲ Het logo toont twee tortelduifjes die een envelop uitwisselen.

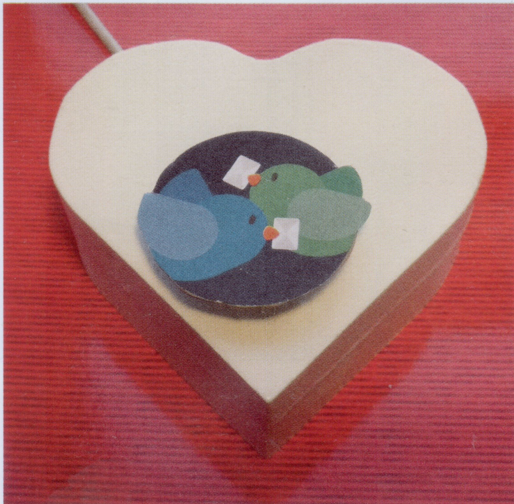


▲ De Raspberry Pi Zero en Raspiaudio MIC+ passen perfect in het doosje.

Voor kinderen en ouders

Olivier besteedde het meeste tijd aan het gebruiksvriendelijk maken van zijn project. "Zo moet je je Raspberry Pi toestemming geven om in jouw naam Telegram-berichten te zenden en te ontvangen. Ik draai daarom een webserver met Flask op de Pi. Die toont je een eenvoudige webinterface voor de authenticatie bij Telegram en om de bevestigingscode in te typen. Uiteraard dien je dan wel het ip-adres van je Pi te kennen of je Pi via de url **raspberrypi.local** te kunnen bereiken; op Windows werkt dat standaard niet."

Hoewel Olivier zijn project bouwde met het idee dat koppels het zouden gebruiken, blijken kinderen tot zijn verbazing meer interesse te hebben in Love Birds: "Twee van mijn vrienden met jonge kinderen maakten het al als een weekendproject. De kinderen gebruiken het thuis om hun ouders te bereiken wanneer die het huis uit zijn. Het is heel educatief om dit project samen als ouder en kind te maken. En het is zo eenvoudig te gebruiken dat zelfs een vierjarige binnen de minuut begrijpt hoe het werkt. En ouders houden van het project omdat het geen scherm heeft met inhoud die mogelijk niet geschikt is voor kinderen. Love Birds is veiliger dan een telefoon en toch leuk en nuttig om te gebruiken."

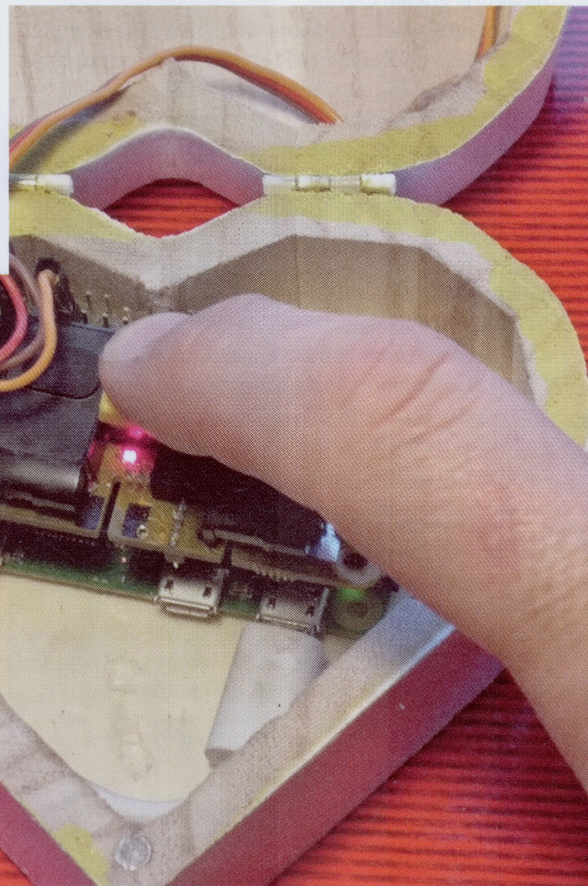


▲ Het doosje is mooi genoeg om in je woonkamer te staan.

Ga zelf aan de slag

Love Birds was niet het werk van Olivier alleen. Zijn vader Louis Ros hielp met de code, zijn vriend Patrick met het debuggen, Line De Carné ontwierp het logo en Roxane Gataud koos de lettertypes. Camille en Edouard bedachten de ideeën achter de video waarin het project voorgesteld wordt en zijn vriendin Livia waarmee alles begon acteerde in de video. Love Birds werd ontwikkeld in het FabLab VilletteMakerz in Parijs.

Wil je zelf je eigen Love Birds maken, dan vind je Oliviers uitleg inclusief video op **raspiaudio.com/lovebirds**. Zijn code kun je bekijken op **github.com/kheperV3/LoveBirds**.



▶ Met een druk op de knop speel je het ontvangen spraakbericht af of spreek je zelf een bericht in.

In een **oogopslag**

- Love Birds is een project om audioberichten tussen twee mensen uit te wisselen zonder extra apparatuur.
- Een Raspberry Pi Zero met audio-dac, luidsprekers en microfoon speelt je binnenkomende berichten af en neemt je antwoorden op.
- Twee van deze apparaten zijn met elkaar verbonden via internet. Of je bouwt er één die via Telegram met een telefoon communiceert.
- Love Birds gebruikt de veilige berichtendienst Telegram, die beschikbaar is op elke telefoon, het web, Linux, ...

Een koelkast voor je schildpad

Stefan Wollners schildpad geniet jaar na jaar van een knusse winterslaap. **Rosie Hattersley** ontdekt dat dit te danken is aan een koelkast die door een Pi aangestuurd wordt.



MAKER

Stefan Wollner

Stefan werkt voor een grote verzekeringsmaatschappij. Hij houdt van technologie en hij leerde zichzelf programmeren.

magpi.cc/dioTvm

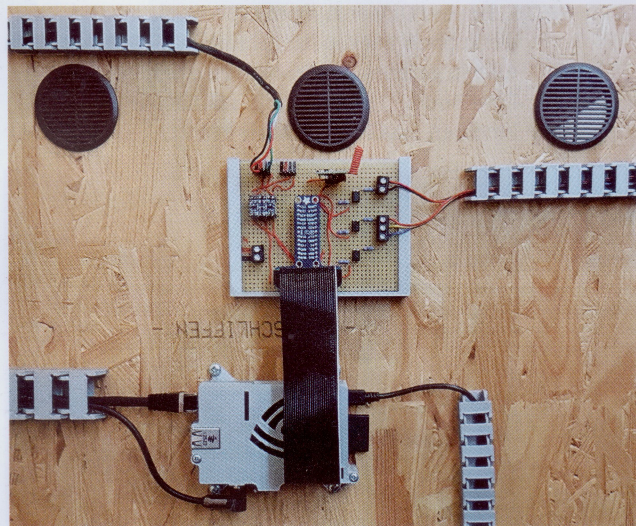
Seizoensgebonden temperatuurverschuivingen zijn voor elk levend wezen een uitdaging. Wij mensen kunnen eenvoudigweg een trui of jas aan- of uitdoen, maar onze huisdieren hebben die optie niet. Technologie-enthousiasteling Stefan Wollner stootte tegen dat probleem aan toen hij in 2015 verhuisde en zich daarna realiseerde dat de kelder te warm was voor zijn schildpad Pumba. Stefan plande toen om zijn schildpad 's winters in een koelkast te laten doorbrengen — iets wat eigenaars van schildpadden wel vaker doen. Maar de temperatuur van een standaard koelkast kun je niet continu monitoren en de interne thermostaat biedt maar een beperkte temperatuurregeling. De warmste en koudste instellingen van een doorsnee koelkast verschillen maar drie graden Celsius.

Toen kreeg Stefan een ingeving: hij realiseerde zich dat hij met een Raspberry Pi en een temperatuursensor over de omgeving van Pumba kon waken. Voor slechts enkele euro's kocht Stefan een DS18B20-temperatuursensor om in de koelkast te steken. Het systeem controleert de temperatuur elke minuut en slaat de resultaten op in een database, die het dan als een realtime grafiek toont. Een Raspberry Pi reageert op de bevindingen van de sensor door met een schakelrelais van 12 V naar 230 V een koelcompressor in of uit te schakelen.

Twee kleine door een Pi aangestuurde schermpjes tonen of alles werkt zoals het hoort. Eén scherm van twee regels toont de huidige temperatuur van de koelkast en van de kelder; het andere toont het tijdstip van de laatste opening van de koelkast en de

▼ Een verhuizing deed Stefan Wollner nadenken over hoe hij zijn nieuwe omgeving kon aanpassen voor de winterslaap van zijn schildpad.







► Een Raspberry Pi van de eerste generatie vormt het hart van zijn installatie.

Het centrale onderdeel van het systeem is een Raspberry Pi.

Alle sensoren en schakelaars zijn met deze printplaat verbonden.

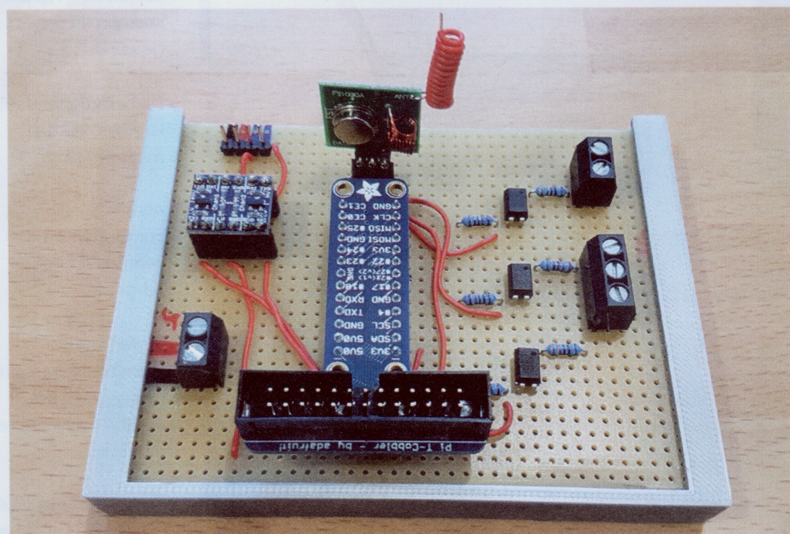
In een **oogopslag**

- Stefan heeft al drie jaar aan dit project gewerkt.
- De tweedehandse koelkast kostte maar 30 euro.
- Allerlei componenten komen uit een 3d-printer.
- Stefan werkt momenteel aan een weegschaal om Pumba's gewicht te monitoren.
- Op het verlanglijstje van Stefans slimme koelkast staat ook nog een ultrasone verstuiver.

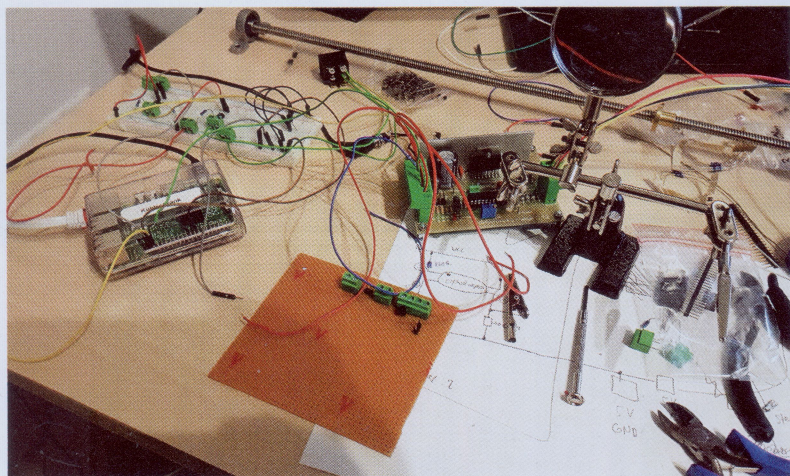
Deze schakelaar controleert het openen en sluiten van de deur.

huidige status van de koelcompressor.

Elke keer dat er iemand in de kelder komt, wordt het backlight van beide schermen door een PIR-bewegingssensor geactiveerd. Maar Stefan hoeft niet naar de kelder te gaan om zijn schildpad te controleren: een andere Raspberry Pi toont de gegevens op twee extra lcd-schermen in de keuken. Als Stefan aan het koken is en wil weten hoe het met Pumba's omgeving is, drukt hij eenvoudigweg op een knop en krijgt hij een update te zien.







▲ Een T-Cobbler Plus break-outboard verbindt de Raspberry Pi met de andere elektronica.

▶ Pumba kauwt graag op bloemen.

▼ Een prototype van het project.

Een hap frisse lucht

De temperatuurcontrole was niet de enige uitdaging waarmee Stefan en zijn geliefde schildpad te maken kregen. Hij moest ook rekening houden met de zuurstoftoevoer. Als Stefan thuis was, opende hij de deur van de koelkast regelmatig, waardoor de zuurstofvoorraad ververst werd. Maar werk en sociale verplichtingen hielden hem vaak weg van huis. Stefan wilde niet riskeren dat de zuurstofvoorraad op zou raken. Daarom creëerde hij een automatisch deuropeningssysteem met een lineaire motor die de koelkastdeur op vastgelegde tijdstippen opent en sluit en controleert of de deur goed gesloten is.

> “Het volgende op mijn verlanglijstje is een ultrasone verstuiver om de luchtvochtigheid te regelen”

De ondertussen hele slimme koelkast levert automatisch foutenboodschappen af, waarschuwingen en meldingen, zodat Stefan altijd op de hoogte blijft van mogelijke problemen met de leefomgeving van zijn gepantserde metgezel. Als het nodig is, kan hij zelfs op afstand de temperatuur en de tijdstippen van de opening van de deur aanpassen, om meer of minder zuurstof in het winterslaapcentrum te laten.

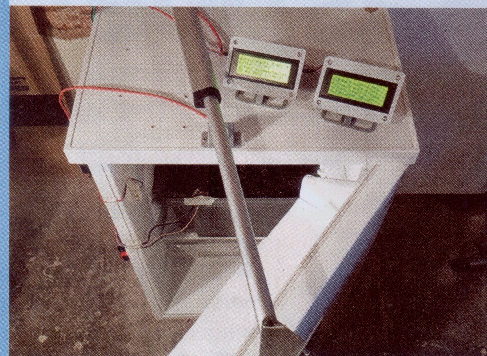
Noodstopprocedures

Stefan heeft talloze uren besteed om de perfecte winterslaapomgeving te maken voor Pumba. De hele keet is ook beveiligd tegen indringers of een stroomuitval. Een Raspberry Pi Zero W beveiligt de hele opstelling. Deze heeft zijn eigen sensor in de koelkast die de temperatuur meet en het systeem volledig uitschakelt als de drempelwaarde bereikt is en een notificatie stuurt.

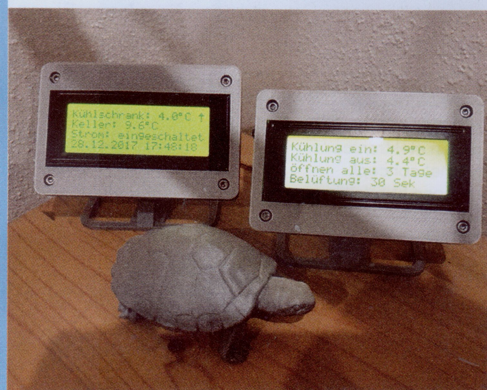
“Waarschijnlijk ben ik nooit klaar met dit project”, zegt Stefan. “Ik ben nu al de volgende verbeteringen aan het plannen. Zo wil ik in de nabije toekomst een ultrasone verstuiver installeren om de luchtvochtigheid te regelen en een weegschaal om het gewicht van Pumba te meten.” Hij wil ook het voederen van de schildpad automatiseren.

De lineaire motor opent en sluit de koelkast op vastgelegde tijdstippen.

Opendeurbeleid



01 Wanneer er niemand thuis is, begint de zuurstofcontroller zijn werk: hij opent de koelkastdeur automatisch op ingestelde tijdstippen om de zuurstofvoorraad te verversen zodat de bewoner blijft ademen.



02 Een heel gamma aan sensoren monitort de temperatuur en het zuurstofniveau in de koelkast en. Lcd-schermen in de kelder en de keuken tonen de huidige status van de koelkast en wanneer de deur het laatst geopend is.



03 Pumba is zich niet bewust van de Herculesarbeid die zijn eigenaar verricht en geniet van de luxe in zijn nest met verse bladeren, nauwkeurig temperatuurgestuurd en met voldoende frisse lucht en voedsel.

Tomografie met een Raspberry Pi

Dit project, dat je via Twitter kunt aansturen, creëert op een heel ongebruikelijke manier 3d-modellen. **Rob Zwetsloot** zoekt uit hoe het werkt.



MAKER
Emma Osbiston

Een student ingenieurswetenschappen in elektronica die het concept met een bundellijn bedacht toen ze bij Diamond Light Source werkte.

magpi.cc/GGQdcd

Tomografie is een proces waarbij elektromagnetische straling met hoge energie (gewoonlijk röntgenstraling) in een voorwerp doordringt en sensoren aan de andere kant er een dwardoorsnede van registreren. Deze doorsnedes worden dan aan elkaar geplakt om een 3d-model van het voorwerp te maken, ook van de binnenkant. Als dat bekend klinkt, heb je waarschijnlijk al wel eens een ct-scan laten uitvoeren — ct staat voor computertomografie. Het is standaardtechnologie in de medische wereld, en tegenwoordig kun je ze zelfs met een Raspberry Pi aansturen.

> De opstelling werkt met een draaitafel om het voorwerp te roteren en de Pi Camera Module neemt een foto van het object in het zichtbare licht van de bundellijn

Emma Osbiston kreeg bij Diamond Light Source, waar men een synchrotron staan heeft, de opdracht om een demonstratie te ontwikkelen van een Pi die zo'n apparaat aanstuurt, maar dan op een veilige manier zonder dat een deeltjesversneller bundels straling wegschiet.

"In essentie kreeg ik de taak om een 'mini-bundellijn' te ontwikkelen met een Raspberry Pi die de data-acquisitiesoftware van Diamond draait", maakt Emma bekend. "Het belangrijkste verschil tussen dit project en een echte bundellijn voor tomografie bij Diamond is dat ik zichtbaar licht moest gebruiken als mijn bundellijn in plaats van röntgenstralen met hoge energie. Dat betekende ook dat ik geen dure, gespecialiseerde detector nodig had, maar eenvoudigweg een Raspberry Pi Camera Module als detector kon inzetten."



▼ Hoe een legomannetje er als een scan uitziet



3d-model

Hoewel veel 3d-opnamesystemen met meerdere camera's werken om talloze foto's tegelijk te maken, werkt deze opstelling met een draaitafel om het voorwerp te roteren. De Pi Camera Module neemt dan een foto van het object in het zichtbare licht van de bundellijn. De software construeert uit deze gegevens een 3d-model.

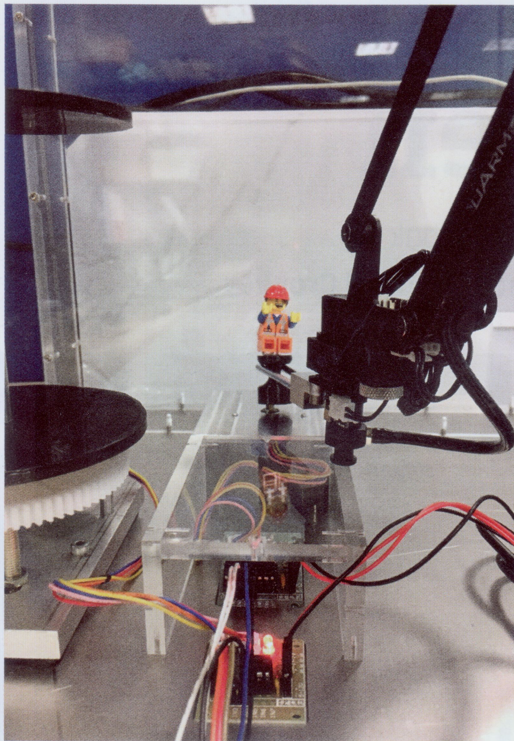
"Het idee van het project was om iets te creëren dat nuttig was om de bundellijnen van Diamond uit te leggen," zegt Emma. "We wilden er ook de flexibiliteit van Diamonds software GDA (*Generic Data Acquisition*) voor synchrotrons mee demonstreren door het op een Raspberry Pi te draaien!"

3d met vrienden

Dat de opstelling via Twitter-berichten aangestuurd kon worden, was een belangrijk aspect van het project. Gebruikers kunnen naar **@DiamondRPi** tweeten met een getal tussen 1 en 10. Een robotarm selecteert dan het gevraagde monster en zet het op de draaitafel voor een scan. De gebruiker waarvan de aanvraag kwam, krijgt daarna de scan in een bericht.

"Ik had graag nog de reactie op Twitter verbeterd", geeft Emma toe. "Bijvoorbeeld door beelden met een hogere kwaliteit te leveren of zelfs geanimeerde gifs, en een voortgangsrapport van de Pi Camera dat het monster toont terwijl het gescand wordt — aan dat laatste wordt momenteel bij Diamond nog gewerkt!"

Maar het project voldeed bij voltooiing tenminste aan de originele specificaties, vertelt Emma ons: "Tegen het einde van de zomer hadden we een succesvol werkende 'bundellijn' die alle originele doelstellingen haalde en zelfs nog meer. Zo was de robotarm oorspronkelijk bedoeld als een extra doelstelling als het project goed zou lopen, en op het einde hadden we die ook succesvol werken! Een gebruiker kon daardoor een verzoek tweeten naar **@DiamondRPi** en de scangegevens terugkrijgen."





In een **oogopslag**

- De volledige naam van het project is: "R03: A Raspberry Pi Tomography Beamline Controlled Through Twitter".
- Door een Pi te gebruiken is het ontwerp reproduceerbaar.
- De Twitter-account **@DiamondRPi** is nog altijd actief.
- Het project werd tijdens een zomerstage van drie maanden uitgewerkt.
- Twitter en de robotarm zijn niet nodig om het project zelf te maken.

Via Twitter wordt een voorwerp gekozen voor de scan.

De Raspberry Pi scant het object met een Camera Module en creëert een 3d-model van het object.

De robotarm pikt het gekozen model en plaatst het op een draaitafel.

Heverlee Sjoelen

Hef een glas op Grant Gibson, voor wie een poging om een sjoelbak te vereenvoudigen tot een verfrissende beloning leidde, zo ontdekt **David Crookes**.



MAKER

Grant Gibson

Grant houdt ervan om fysieke projecten te maken. Hij heeft een aantal interactieve games, kiosks en museumexposities geproduceerd.

magpi.cc/DQiGwQ

Elke Nederlander of Belg is wel bekend met sjoelen. Je weet dan dat het doel van dit spel eenvoudig is — houten schijven door de vier kleine gleufjes op het einde van de plank schuiven — maar dat de regels nogal ingewikkeld zijn.

Het lijkt misschien wat vreemd dat een spel dat zoveel op oog-handcoördinatie en punten tellen vertrouwt de perfecte match is voor een project in opdracht van een biermerk. En toch heeft Grant Gibson succes met zijn verfrissende interpretatie van sjoelen. Hij vereenvoudigde de regels en voorzag de sjoelbak van een Raspberry Pi om speciale prijzen aan de winnaars te serveren.

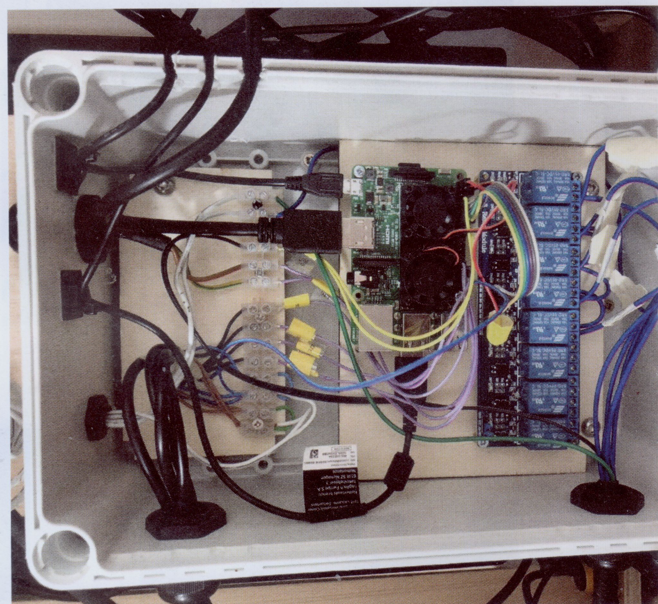
"De traditionele puntentelling van sjoelen vraagt heel wat optellingen en vermenigvuldigingen, maar onze versie geeft spelers eenvoudigweg tien schijven en vereist dat ze drie daarvan binnen dertig seconden door gelijk welke van de vier gleufjes schuiven", legt Grant uit.

Terwijl de speler dit doet, registreert de Pi (een Model 3B) hoeveel schijven er door elke gleuf schuiven, houdt hij bij hoeveel tijd de speler nog over heeft en toont hij een boodschap op het scherm als de speler wint. Door een Logitech HD-webcam kunnen toeschouwers onmiddellijk de reactie van de speler zien, die snel tussen frustratie en succes wisselt. "Sjoelen is een heel tastbaar spel met een lange, interessante geschiedenis", merkt Grant op.

Geïnspireerd door een flipperkast

Grant startte zijn project met enkele doelen in zijn achterhoofd: "Ik wilde iets wat je in een kleine bestelwagen kon transporteren en met twee personen kon monteren. Ik wilde ook dat het er *vintage* uitzag." Geïnspireerd door flipperkasten, kwam hij op het idee van een montage in drie delen, die je plat kunt leggen voor het transport en dan snel op de plaats zelf kunt monteren.

De Pi 3B bleek de perfecte computer voor dit project te zijn. Hoewel Grant altijd al snel de neiging had om pc's op volledige grootte te gebruiken in zijn projecten, zegt hij dat de Pi hem toeliet om de software minder complex te maken: hij kon de Pi gewoon met een schermvullende Chromium-browser via json laten communiceren om de puntentelling af te handelen en taken in JavaScript te tonen. En door de Pi had hij ook geen externe hardware meer nodig om de I/O aan te sturen. Hij kon immers eenvoudig Python gebruiken voor de I/O-taken.



▲ Om het mechanisme van de bierautomaat te laten werken, stuurt de Pi via zijn gpio-pennen relais aan die op hun beurt actuators voor een autodeurvergrendeling activeren.

"Infraroodsensoren detecteren wa nneer er een schijf door een gleuf schuift", voegt Grant toe. "Door de snelheid van de schijven moesten we alle vier de infraroodsensoren meer dan 100 keer per seconde uitlezen om zeker te zijn dat we de schijven altijd detecteren. De Python-code optimaliseren om snel genoeg te draaien en tegelijk voldoende rekenkracht over te laten om een schermvullende webbrowser en HD-webcam aan te sturen was zeker de grootste software-uitdaging in dit project."

Schuimende bierfontein

De Raspberry Pi gebruikt zijn gpio-pennen om een blikje Heverlee-bier aan de winnaar te geven. Die blikjes zitten in de machine, maar Grant brak zijn tanden stuk op het mechanisme van de bierautomaat: het moest lichtgewicht zijn, compact en de blikjes koel houden.

Geen enkele commercieel beschikbare automaat had hiervoor een oplossing. Grants initiële

De kast heeft een ingebouwde bierautomaat, geverfd in het blauw van het Belgische biermerk Heverlee.

Het scherm toont het beeld van de HD-webcam samen met animaties die de punten bijhouden.

Grant bouwde zijn sjoelbak met een standaardplank van Masters Traditional Games en hij besteedde al een boel lunchtijden aan het spel.

Sjoelen is al een tijdje weer hip. Het is een perfect spel voor in de kroeg.

pogingen met stappenmotoren en lasergesneden acryl tandwielen waren rampzalig. "Na een tiental succesvolle afleveringen van een blikje, was het prototype niet meer uitgelijnd en begon het blikjes door te snijden, wat een enorm schuimende fontein van bier opleverde. Indrukwekkend om te zien, maar geen geweldige combinatie met elektronica", lacht Grant.

In de plaats daarvan tekende hij een finaal ontwerp, lasergesneden uit triplex van populierenhout. "Het gebruikt motoren voor een centraal vergrendelingssysteem in een auto om met een wipmechanisme de blikjes op te dienen. Een aangepaste warmtewisselaar met Peltier-effect en wat bij elkaar geraapte pc-ventilatoren houden de blikjes in de machine koel", onthult Grant zijn geheim.

"Ik zou nu nog eens een keer een lichtgewichtversie willen maken, misschien met een opvouwbare sjoelbak en een scorebord dat ineens tevoorschijn komt, en door één persoon gedragen kan worden", voegt hij toe. Daar zouden we zeker op drinken.

In een **oogopslag**

- Sjoelen is populair in Nederland, België en Duitsland.
- Grant had 150 uur nodig om deze versie te maken.
- De totale kostprijs van de onderdelen bedroeg minder dan € 470.
- Een lokale meubelmaker bouwde de kast.
- De winnaars krijgen een blikje bier.



Bouw je eigen telefooncentrale

Transformeer je simpele telefoonlijn thuis tot een telefooncentrale tjokvol functies. De benodigdheden? Een Raspberry Pi en Asterisk.



MAKER

PJ Evans

PJ is auteur, software-ingenieur en organisator van een Raspberry Jam. Je telefoontje is belangrijk voor hem.

mrpjevans.com

@mrpjevans

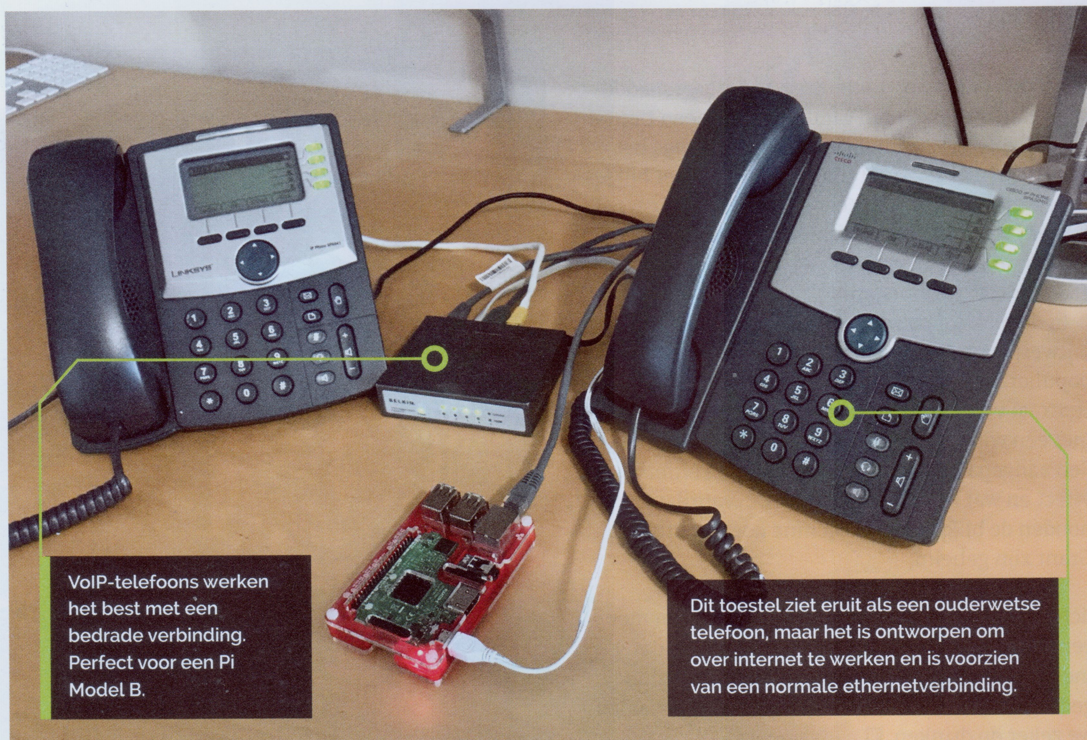
Je hebt nodig

- 2 × VoIP-telefoons, zoals de Cisco SPA504G.
- RasPBX-image **magpi.cc/tUHyFb**
- Ethernetkabels en switch
- Android- of iOS-toest el (optioneel)

Een traditionele telefooncentrale (*Private Branch eXchange* of *PBX*) is essentieel voor elk bedrijf, maar is niet goedkoop. Zelfs een eenvoudig systeem kan duizenden euro's kosten om te implementeren. Maar met de hulp van VoIP (*Voice over Internet Protocol*) en het populaire opensource telecomplatform Asterisk kan onze favoriete Raspberry Pi dezelfde functies aanbieden als de duurste telefooncentrales. Een klein bedrijf heeft zo in combinatie met twee VoIP-telefoons voor minder dan € 120 een telefooncentrale, inclusief alle leuke functies. En waarom installeer je thuis geen telefoons in je studeerkamer of je keuken? Dan kun je een voicemailbericht voor je hond inspreken!

01 Kies je toestel

Een van de geweldige eigenschappen van VoIP-telefoonsystemen is hun flexibiliteit: je bent niet beperkt tot een traditionele telefoon. Er zijn drie belangrijke opties. We gebruiken de populaire Linksys SPA941 en Cisco SPA504G SIP-telefoons. Dit zijn al oudere toestellen, dus met wat geluk vind je ze voor niet veel meer dan een tientje op een tweedehandswebsite. Er zijn nog talloze andere types IP-telefoons beschikbaar, zoals de draadloze telefoons met DECT. Een andere optie is om een traditionele PSTN-telefoon met een ATA-omzetter



VoIP-telefoons werken het best met een bedrade verbinding. Perfect voor een Pi Model B.

Dit toestel ziet eruit als een ouderwetse telefoon, maar het is ontworpen om over internet te werken en is voorzien van een normale ethernetverbinding.



Je hebt een ethernetswitch nodig — als je geen poorten meer over hebt, kies dan voor VoIP-telefoons met doorvoerconnector om andere toestellen aan elkaar aan te sluiten.

voor VoIP geschikt te maken — hoewel je dan wat functies verliest. Tot slot kun je een softphone zoals Zoiper op een Android- of iOS-toestel of op je computer draaien.

02 Netwerkconfiguratie

VoIP-systemen vertrouwen op een snel netwerk met weinig vertraging om goede audiokwaliteit te kunnen aanbieden. Daarom hebben veel VoIP-toestellen, inclusief onze keuze van telefoons voor dit artikel, geen wifi. Voor de beste prestaties gaan we alles op de 'ouderwetse manier' aansluiten met een kleine ethernetswitch. Veel VoIP-telefoons zijn uitgerust met een doorvoerconnector, zodat je er andere toestellen op kunt aansluiten zonder alle poorten van je switch te bezetten. Je kunt ook instellen dat de telefoon prioriteit krijgt over dataverkeer, zodat je telefoontjes nog altijd werken terwijl je de nieuwste Fortnite-update aan het downloaden bent — die functie heet QoS (*Quality of Service*).

03 Bereid je telefoons voor

Als je je telefoons tweedehands hebt aangekocht zoals wij deden, dien je ze waarschijnlijk eerst te resetten voordat je ermee aan de slag kunt. VoIP is niet één protocol, maar bestaat uit meerdere protocols die samenwerken en de mogelijke instellingen zijn dan ook immens. Om later frustraties te vermijden, is het van vitaal belang om elke telefoon volledig tot zijn standaardinstellingen te resetten en te testen of hij succesvol met het netwerk verbonden raakt. Op ons model ga je naar **Setup** (de toets met het papier) en dan **Factory Reset**. Laat je telefoon herstarten en controleer dan **Setup > Network**.

> Om later frustraties te vermijden, is het van vitaal belang om elke telefoon volledig tot zijn standaardinstellingen te resetten

Als alles goed is, krijg je nu het ip-adres van je telefoon op het scherm te zien.

04 Installeer RasPBX

Asterisk is populaire en volwassen software die een traditionele telefooncentrale implementeert. De mogelijkheden en functies zijn duizelingwekkend, zeker als je weet dat het om een opensourceproject gaat. Het nadeel is dat Asterisk wat lastig kan zijn om te configureren. Gelukkig is er RasPBX, een Raspbian Stretch-gebaseerde distributie met Asterisk en de webgebaseerde front-end FreePBX voorgeïnstalleerd. Download het image van **raspberry-asterisk.org** en schrijf het naar een microSD-kaart (wij gebruikten

Tip 

Prestaties zijn belangrijk

VoIP-audiocodecs vertrouwen op een goed netwerk. Wil je de beste prestaties, kies dan voor een snelle switch.



▲ VoIP-telefoons zoals deze Linksys SPA941 zijn gemakkelijk op tweedehandswebsites te vinden en bieden talloze functies.

balenaEtcher van **www.balena.io/etcher**). De instructies voor de installatie vind je op **magpi.cc/pgmfnY**.

05 Stel FreePBX in

Zodra de basisconfiguratie in orde is, open je **http://raspbx.local/** in je webbrowser. Als dat niet werkt, zoek dan het ip-adres van je Pi op met een app zoals Fing op Android of iOS en vul het gevonden ip-adres in je webbrowser in. FreePBX start nu op en leidt je door een eenvoudig configuratieproces waarin je een beheerdersaccount voor het systeem aanmaakt. Wanneer dat gebeurd is, lijkt het aantal opties en schermen dat je te zien krijgt wat verwarrend, maar maak je heen zorgen: de standaardinstellingen van RasPBX zijn verstandig en we hoeven daarom verder niets te veranderen om aan de slag te gaan.

06 Creëer extensies

In VoIP/PBX-terminologie heet elk eindpunt voor een telefoontje een extensie in plaats van een telefoon, omdat de meeste VoIP-telefoons meerdere extensies kunnen afhandelen. Laten we het in dit artikel eenvoudig houden met één extensie per telefoon. Klik in de webinterface van FreePBX op **FreePBX Administration**, log in als beheerder, klik op **Applications** en dan op **Extensions**. Klik op die pagina op **Quick Create Extension**. Kies een uniek extensienummer (we kozen met veel fantasie 1) en een weergavenaam (de gebruiker). Herhaal dit voor alle extensies die je wilt, en vergeet niet op **Apply Config** te klikken om je wijzigingen door te voeren.

07 Configureer de telefoons

De configuratie van je telefoons hangt van het model af. Belangrijk om te weten is het ip-adres van de Pi, het extensienummer en het wachtwoord (of secret). Om dit wachtwoord te weten te komen, klik je op het **Edit**-icoontje van een van de extensies die je gecreëerd hebt. Zoek nu het ip-adres van de telefoon op, bijvoorbeeld met Fing. De kans is groot dat je telefoon op dat ip-adres een webinterface heeft. Toen wij het ip-adres van onze telefoons in onze webbrowser bezochten, konden we op **Admin Login** klikken en dan **Ext 1**. We voegden het ip-adres van de Pi bij **Proxy** in, de extensie bij **User ID** en het wachtwoord bij **Password**. Na een reboot zouden je telefoons zich bij RasPBX moeten registreren.

08 Bellen maar!

Bij de telefoons van Linksys en Cisco leidt een succesvolle registratie tot groene lampjes, die aanduiden dat de telefoon klaar is om oproepen te ontvangen. Veronderstel dat je extensies 1 en 2 hebt. Neem de hoorn van extensie 1 op, kies 2 op het toetsenblok en druk op de knop om te bellen. Als alles correct geconfigureerd is, zou je andere telefoon nu moeten rinkelen. Laat iemand anders de hoorn van die telefoon opnemen en je beantwoorden (of praat tegen jezelf, we zullen het aan niemand vertellen) en controleer of de audio in beide richtingen in orde is. Je hebt nu een telefooncentrale! En laat een van de telefoons eens van de haak en bel hem op: je krijgt dan een voicemailbericht.

09 Voicemail

Een van de meest voor de hand liggende functies die we nodig hebben, is voicemail. Ook daar komt RasPBX van pas. Zonder enige verdere configuratie heb je al een volledig voicemailsysteem op je Pi draaien! Daartoe krijg je toegang met een 'functiecode': een nummer met een speciale betekenis dat je door een * laat voorafgaan. De functiecode voor voicemail is *97. In het menu van onze telefoon konden we dat koppelen aan de knop voor voicemail, zodat het gebruik nog eenvoudiger wordt. De eerste keer dat je belt, word je door de instellingen geleid zodat

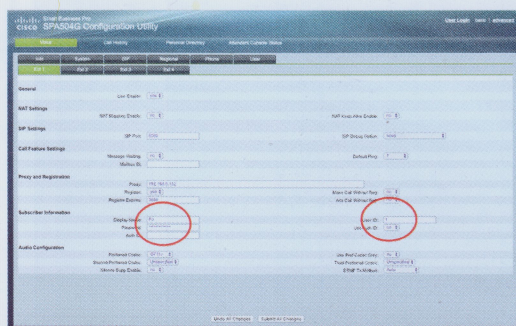
je een pincode en persoonlijke boodschap kunt toekennen. Als je er nog wat meer aan sleutelt, kun je zelfs de ingesproken boodschappen als bijlage naar je laten e-mailen.

10 Functies, functies, functies

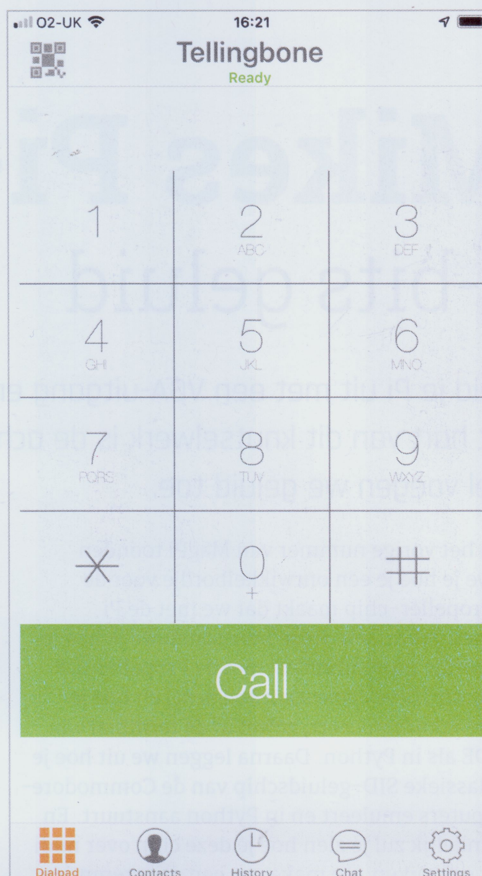
Nu je de basis hebt getest, heb je genoeg om mee te spelen. Standaardfuncties van telefooncentrales zoals oproepen doorverbinden, oproepgroepen (teleconferenties) en een niet-storen-functie die inkomende oproepen onmiddellijk naar je voicemail doorstuurt, het werkt allemaal al standaard. Kijk maar eens rond in de webinterface van FreePBX, in het bijzonder de sectie **Applications**. Je vindt er nog talloze andere krachtige functies. Probeer eens *call recording* (opnames van oproepen), *ring groups* (waar een inkomende oproep van extensie naar extensie wordt doorverbonden tot iemand opneemt of zelfs meerdere extensies tegelijk doet rinkelen), en zelfs *interactive voice response* (IVR, "druk op één voor dit, druk op twee voor dat"). En als dat nog niet genoeg voor je is, zijn er nog geavanceerdere functies beschikbaar als commerciële uitbreidingen.

11 Soft phones

We vermeldden in het begin van dit artikel al dat VoIP-telefoons niet de enige optie zijn. Er bestaan ook heel wat goede softphones. Zoals de naam al suggereert, gaat het hier volledig om software. Die draait op je smartphone, tablet of computer. Een populaire keuze is Zoiper, dat voor Windows, macOS, Linux, Android en iOS bestaat. Na installatie voeg je een nieuwe account toe en kies je voor een manuele instelling. Vul je extensienummer als de gebruikersnaam in, het bijbehorende wachtwoord, en het ip-adres van je Pi als het domein. Klik op **Register** en binnen enkele seconden is je 'telefoon' online.



Stel je VoIP-telefoon in: het gebruikers-id en wachtwoord



Je Raspberry Pi-telefooncentrale werkt ook met softphones zoals Zoiper, dat op talloze toestellen draait.

12 Trunking

Om oproepen naar buiten mogelijk te maken en oproepen van buitenaf te kunnen ontvangen, heb je een provider nodig die je oproepen naar het publieke telefoonnetwerk kan doorsturen en inkomende oproepen naar jou doorstuurt. Dat staat bekend als een SIP-trunk. Op websites zoals **www.voip-info.org** vind je lijsten met VoIP-providers die een SIP-trunk aanbieden. Het vraagt wat configuratiewerk om je telefooncentrale met een SIP-trunk te laten werken, maar een goede VoIP-provider biedt daarvoor gedetailleerde instructies. Daarna kun je met iedereen overal in de wereld bellen via je telefooncentrale.

13 Uitbreidingen

Je kleine Raspberry Pi kan nog meer. Sommige geavanceerde uitbreidingen voor Asterisk zijn goed ondersteund door RasPBX. Zo kun je een lijst met nummers toevoegen waarvan je oproepen blokkeert, je kunt een faxfunctie toevoegen die je met je e-mail koppelt, en je kunt zelfs een 3G/4G gsm usb-dongel aansluiten om oproepen over het mobiele netwerk door te sturen als je vaste internetverbinding uitvalt. Er is echt heel weinig wat een dure telefooncentrale kan doen dat RasPBX op je kleine Raspberry Pi niet voor een fractie van de prijs kan.

Tip

Geen verschil?

Als je veranderingen in FreePBX doorgevoerd hebt en die geen effect lijken te hebben, klik dan op **Apply Config** in de rechterbovenhoek.

Deel 02

Mikes Pi-bakkerij:

8-bits geluid

Breid je Pi uit met een VGA-uitgang en een retro 8-bits geluidsemulator. Het hart van dit knutselwerk is de achtkerns Propeller-chip. In dit tweede deel voegen we geluid toe.



MAKER

Mike Cook

Een oudgediende tijdschriftauteur en schrijver van de serie Body Build. Coauteur van *Raspberry Pi for Dummies*, *Raspberry Pi Projects* en *Raspberry Pi Projects for Dummies*.

magpi.cc/TPaUfT

In het vorige nummer van MagPi toonden we je hoe je een ontwikkelbordje voor de Propeller-chip maakt dat we met de Pi kunnen gebruiken. Deze keer tonen we je hoe je het bordje programmeert met de Propeller IDE (*integrated development environment*) en hoe je ermee communiceert, zowel in de terminal van de IDE als in Python. Daarna leggen we uit hoe je de klassieke SID-geluidschip van de Commodore-computers emuleert en in Python aanstuurt. En uiteindelijk zul je zien hoe je deze chip over midi kunt gebruiken: we maken er een driestemmige polyfone synthesizer van.

01 Installeer de IDE

Eerst dien je de IDE voor de Propeller-processor te installeren. Open een Terminal en typ de volgende opdrachten in:

```
wget https://github.com/parallaxinc/PropellerIDE/releases/download/0.36.7/propelleride-0.36.7-armhf.deb
sudo apt-get install libqt5serialport5
sudo dpkg -i propelleride-0.36.7-armhf.deb
```

Daarna vind je de Propeller IDE terug in het applicatiemenu van Raspbian onder het submenu **Programming**. Klik erop om het scherm van **afbeelding 1** te krijgen. Om de installatie te voltooien, selecteer je een van de elementen in het hulpmenu, waarna je de vraag krijgt om een toepassing te kiezen om pdf-bestanden te lezen. Selecteer de pdf-lezer in het submenu **Utilities**. Kies dan in het menu **Edit / Preferences** van de IDE het tabblad **theme** en selecteer een thema dat je ligt — zie **afbeelding 2**.

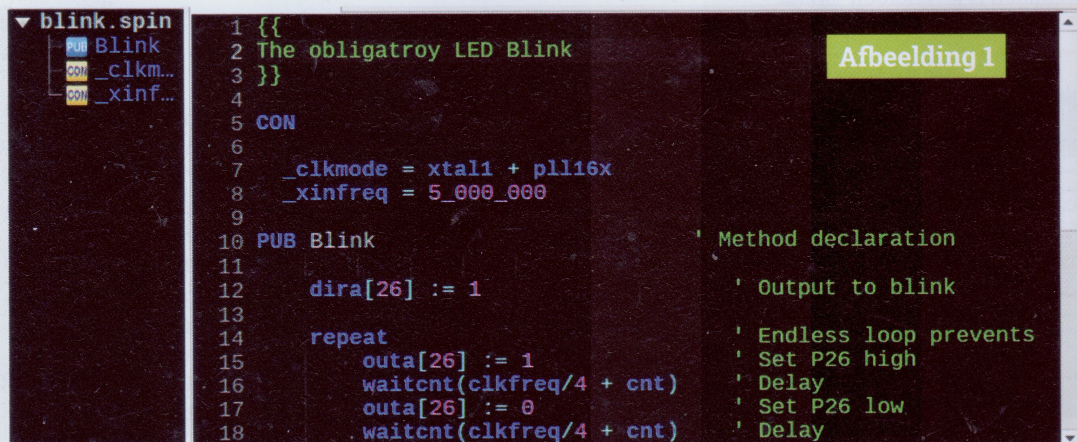
02 Je eerste programma

Het eerste wat je in bijna elk project rond *physical computing* doet, is een led laten knipperen. We hebben een led op pin 26 van ons ontwikkelbordje aangesloten en we gaan die doen knipperen. Klik daarom op het icoontje **New File** — het meest linkse op de iconenbalk bovenaan de IDE. Typ dan de code **blink.spin** in die je achteraan dit artikel vindt. Klik daarna op het icoontje van de hamer om te controleren dat de code correct compileert en selecteer als seriële interface in het uitklapmenu **ttyUSB0**. Klik daarna op het pijltje dat naar rechts wijst om de code naar het bordje te downloaden en uit te voeren.

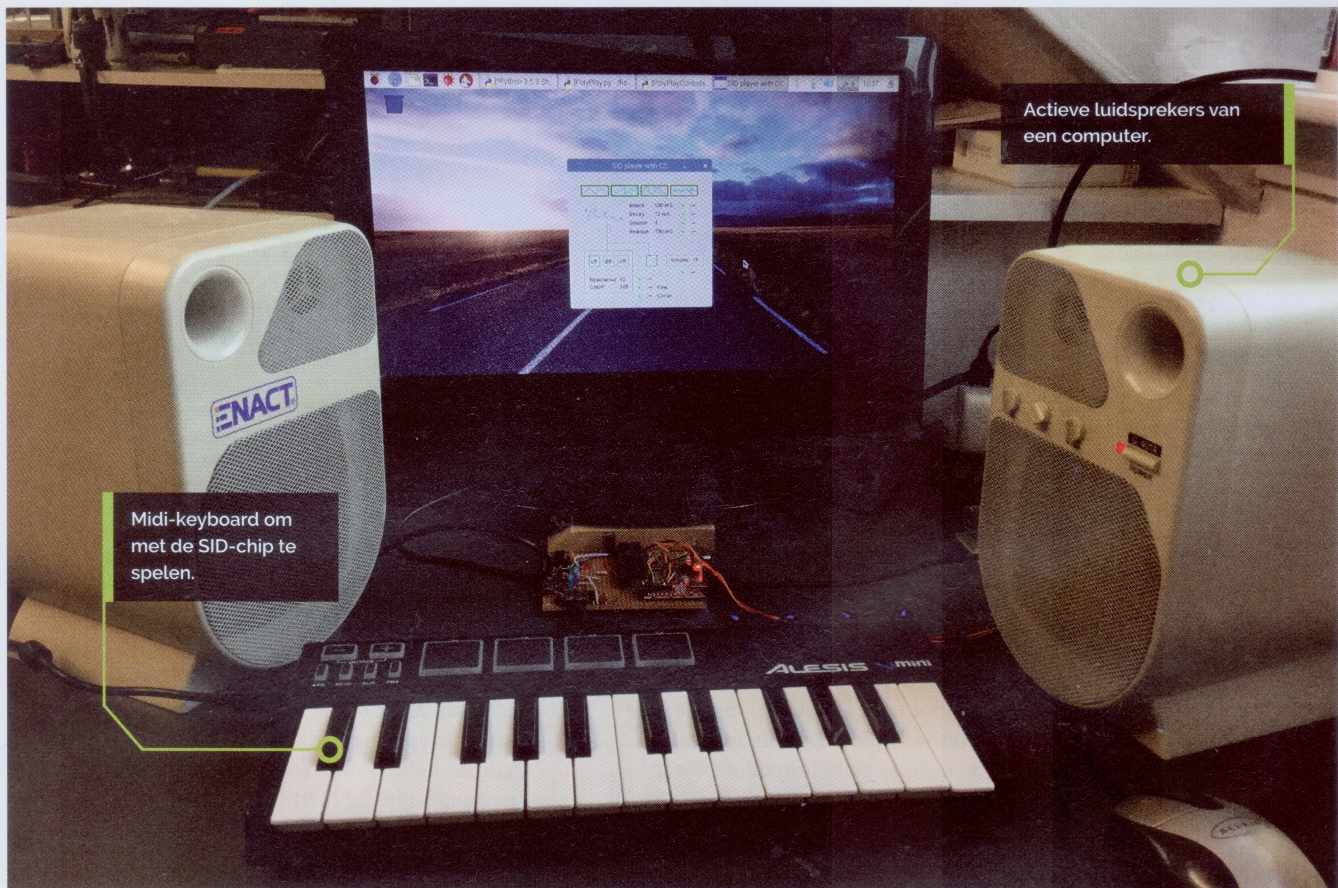
Je hebt nodig

- Propeller-ontwikkelbordje uit MagPi 7

  www.magpi.nl/magazine/2019/7
- Actieve (gevoede) luidspreker
- Midi-keyboard



Afbeelding 1 Ons favoriete kleurenschema voor de IDE



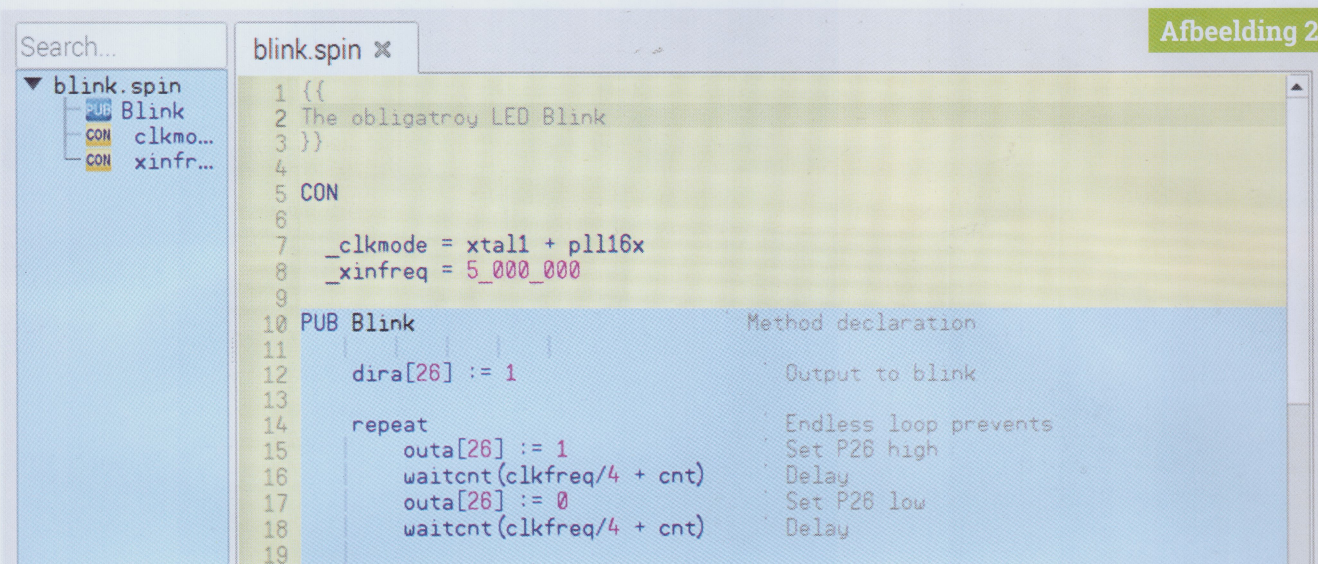
> Deze code is in de programmeertaal Spin geschreven, die in het begin misschien een beetje vreemd lijkt



Een andere actieve luidspreker

03 Spoedcursus Spin

Deze code is in de programmeertaal Spin geschreven, die in het begin misschien een beetje vreemd lijkt, maar niet zoveel verschilt van andere procedurele talen zoals Python. In een Spin-programma definieer je eerst constanten en dan roep je objecten aan die programma's zijn voor andere processorkernen (de cogs). Daarna definieer je functies en uiteindelijk datalijnen. Het gelijkheidsteken (=) is gereserveerd om constanten te definiëren; om waardes aan variabelen toe te kennen gebruik je de operator :=. Die kun je nog het best omschrijven als "wordt gelijk aan". We raden je aan om alle spin-bestanden voor één project in dezelfde directory te plaatsen.



▲ **Afbeelding 2** Een van de andere kleurenschema's voor de IDE

04 Code hergebruiken

Het gebruik van 'cogs' die anderen geschreven hebben, maakt het systeem heel krachtig: daardoor hoef je het wiel niet de hele tijd opnieuw uit te vinden. Een uitstekende bron van cogs is de website Propeller Object Exchange (**magpi.cc/JqaPdV**). En door de pdf op **magpi.cc/sZBzVx** te downloaden, kun je heel wat eenvoudige voorbeelden bekijken. De beste manier om met Spin te leren programmeren, is met de code te spelen en zaken aan te passen. En als je een formelere introductie in Spin wilt, raden we je het boek *Programming the Propeller with Spin* van Harprit Singh Sandhu aan.

05 Communicatie

Het programma **SerialAdd.spin** demonstreert eenvoudige communicatie tussen de Spin-processor en het toetsenbord van de Pi. Het vraagt twee getallen als invoer, telt ze op en toont het resultaat. Deze code roept de cog **FullDuplexSerialPlus.spin** op, die je in onze GitHub-repository of op de Object Exchange vindt. Je dient ze in dezelfde directory te plaatsen als **SerialAdd.spin**. Nadat je met het hamericoontje gecontroleerd hebt of ze compileert, dien je de code deze keer uit te voeren door op het pijltje naar beneden te klikken. Dat laadt het programma in het EEPROM, zodat het bij elke opstart en herstart van het systeem automatisch draait.

06 De terminal van de IDE

Open de terminal van de IDE, selecteer **ttyUSBO** en klik op de tab Reset. Verwijder het vinkje bij het vakje **Echo** en klik dan in het venster van de terminal om er de cursor op te focussen. Wanneer je twee getallen gevraagd worden, typ ze dan in, gevolgd door **RETURN**. Je krijgt dan de som van die twee getallen te zien. Dat herhaalt zich dan. Merk op: je ziet niet onmiddellijk wat je intypt, maar alleen nadat je op **RETURN** gedrukt hebt. Met deze cog kunnen we op een heel gelijkaardige manier gegevens zenden en ontvangen in Python.

07 Geluiden verkennen

Ga naar de Object Exchange en selecteer daar de sectie **Speech/Sound**. Probeer eens wat van de voorbeelden uit. We raden je aan om de zangdemo's van Chip Gracey en 'Joy to the world' zeker eens uit te proberen. Sluit een actieve luidspreker (een luidspreker met ingebouwde versterker) aan op het jack-socket op het bordje.

> " Het programma draait automatisch bij elke opstart en herstart van het systeem "

Een andere aanrader is de Propeller Signal Generator, zeker als je over een oscilloscoop beschikt om de golfvormen te bekijken. Download tot slot **SIDcog v1.3** en voer de code Example uit. Merk op: de code **Dump Play** werkt niet omdat ons bordje niet over een SD-kaartlezer beschikt.

08 De SID-chip

De MOS 6581 SID (Sound Interface Device) is de krachtigste geluidsgeneratorchips die in de vroege generatie computers te vinden is. Hij bestaat

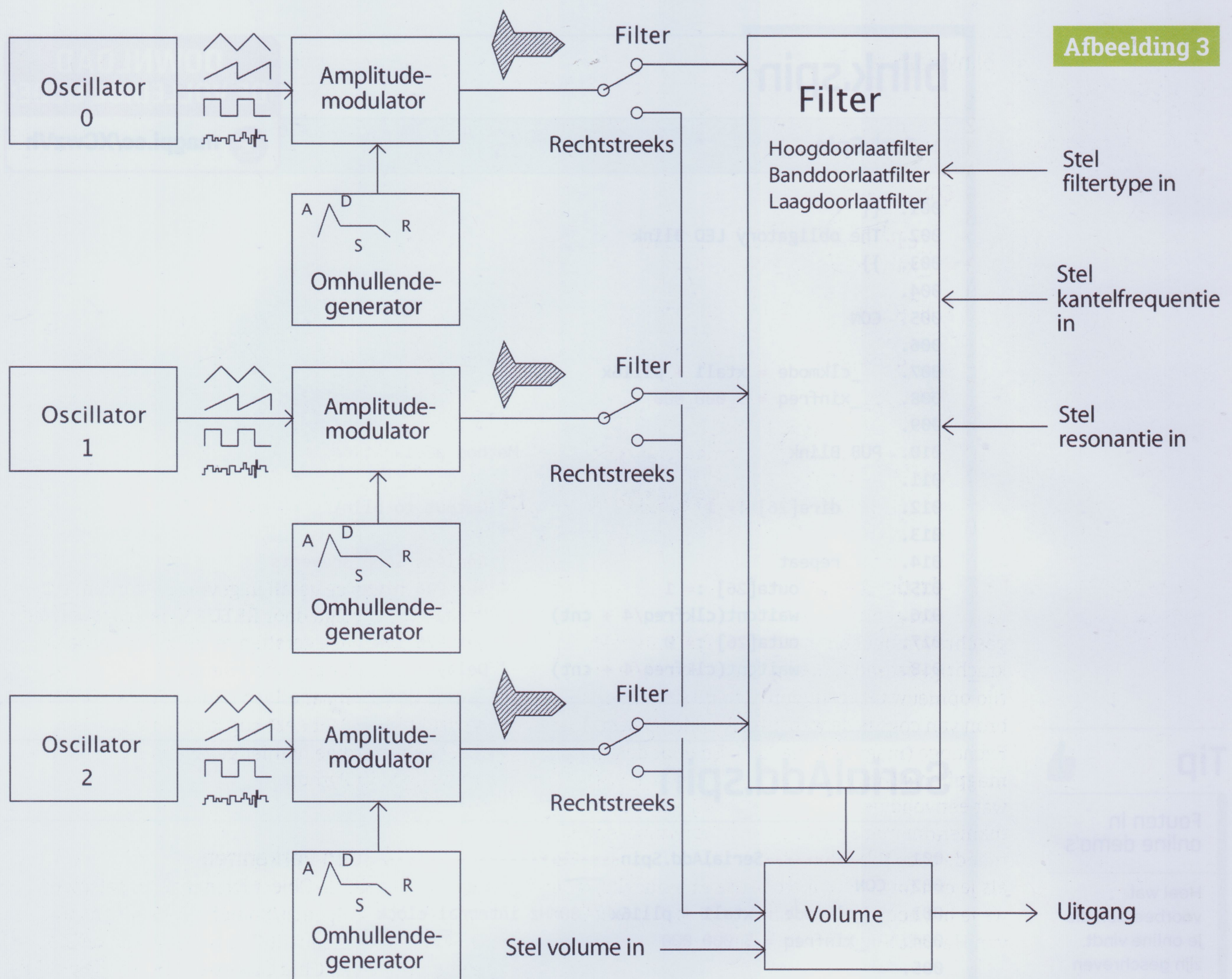
Tip



Standaard seriële poort

De IDE gebruikt een seriële poort. Standaard is dat **ttyAMAo**, wat vroeger de seriële poort van de gpio's was. Maar met de introductie van Bluetooth op de Pi 3-modellen is dat veranderd naar **ttySo**. De IDE pikt die niet op, dus je dient **ttyUSBo** te selecteren.

Afbeelding 3



Afbeelding 3
Blokdiagram van de SID-chip

uit drie golfvormgenerators, elk gevolgd door een ADSR-schakeling (*attack, decay, sustain, release*) om een omhullende te genereren. Een optionele filter kan op elk van deze of alle golfvormen inwerken. De filterparameters zijn door externe componenten aan te sturen, wat exacte emulatie moeilijk maakt. Het blokdiagram vind je in **afbeelding 3**. Download zeker eens een datasheet om de registerstructuur van de chip te bekijken.

09 Met de SID spreken

Muziek is beschikbaar in bestanden met de extensie .sid, maar die zijn complex. Ze bevatten immers niet alleen de gegevens van de song, maar ook de machinecode die nodig is om de juiste gegevens op het juiste tijdstip in de juiste plaats te krijgen. Ze zijn dus wat lastig om te emuleren. We gaan dat hier niet proberen te doen, omdat er al enkele .sid-spelers bestaan en we iets nieuws willen doen. In onze GitHub-repository tonen we voorbeelden van hoe je gegevens in Python naar een SID-cog stuurt. Maar wat we echt willen doen, is de SID-cog bespelen alsof het een midi-geluidsgenerator was.

10 Meerstemmig

Er zijn twee aanpakken mogelijk om een SID-chip met midi te bespelen: eenstemmig of meerstemmig. Of anders gezegd: slechts één noot tegelijk of meerdere noten tegelijk. Elke aanpak heeft zijn voor- en nadelen. Met de eenstemmige aanpak kun je verschillende soorten geluiden maken, maar je kunt geen akkoorden afspelen. We hebben ervoor gekozen om meerstemmig te werken, om een echt speelbaar instrument te kunnen maken. Met drie golfvormgenerators in de chip, hebben we besloten om een driestemmig instrument te maken. De golfvorm, filterschakelaar en ADSR-omhullende van elke oscillator zijn dan hetzelfde.

blink.spin

> Taal: **Spin**

DOWNLOAD DE VOLLEDIGE CODE:


magpi.cc/XCwzVh

```
{{
The obligatory LED Blink
}}

CON

  _clkmode = xtal1 + pll16x
  _xinfreq = 5_000_000

PUB Blink                                  ' Method declaration

    dira[26] := 1                          ' Output to blink

    repeat                                 ' Endless loop prevents
        outa[26] := 1                      ' Set P26 high
        waitcnt(clkfreq/4 + cnt)           ' Delay
        outa[26] := 0                      ' Set P26 low
        waitcnt(clkfreq/4 + cnt)           ' Delay
```

Tip



Fouten in online demo's

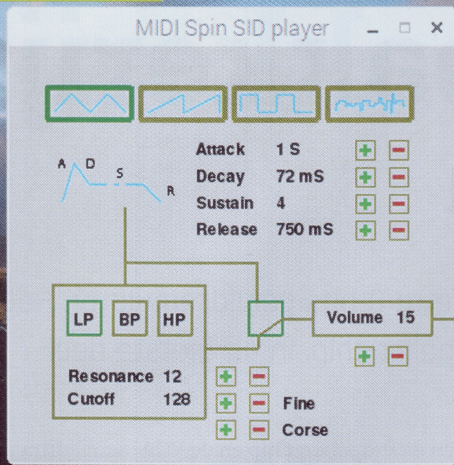
Heel wat voorbeelden die je online vindt, zijn geschreven op een bestandssysteem dat geen verschil kent tussen hoofdletters en kleine letters. Het resultaat is dat je een foutmelding 'File not found' kunt krijgen wanneer je ze probeert te compileren omdat de Pi hoofdletters en kleine letters als verschillend beschouwt. Als je dat voorhebt, verander dan eenvoudig de naam van het betreffende bestand zodat die overeenkomt met de verwachte bestandsnaam.

SerialAdd.spin

```
''----------SerialAdd.Spin-----------------------
CON
  _clkmode = xtal1 + pll16x ' 80MHz internal clock
  _xinfreq = 5_000_000

RxPin = 31      'default boot loader RX
TxPin = 30      'default boot loader TX
BaudRate = 115200 'this is a good baud rate for 80MHz
OBJ
  SerIO : "FullDuplexSerialPlus" 'object that implements serial I/O for the Propeller.

PUB main | N1, N2' The prime function in Cog 0 and first run on boot
SerIO.Start(RxPin, TxPin, %0000, BaudRate) ' initialise SerIO object
SerIO.Str(String("Adding up two numbers ")) ' Introduction text
SerIO.tx(10)                                ' send LF
repeat
   SerIO.tx(10)                             ' send LF
   SerIO.Str(String("Input two numbers "))
   N1 := SerIO.GetDec                       'get dec number i.e. text numeric
   SerIO.Dec(N1)
   N2 := SerIO.GetDec
   SerIO.Str(String(" + "))
   SerIO.Dec(N2)                            'get next one
   SerIO.Str(String(" = "))
   N1 := N1+N2                              'add them
   SerIO.Dec(N1)                            'send back the result as text
```



▲ **Afbeelding 4** De grafische gebruikersinterface van de midi-speler

11 Meerstemmige Pi

We kozen het beproefde framework Pygame om de midi-invoer af te handelen en, nog belangrijker, de meerstemmigheid. Dat is best lastig: we dienen niet alleen uit te zoeken op welk SID-kanaal we een binnenkomende noot moeten afspelen, maar als er geen kanaal vrij is, moeten we eerst een kanaal vrijmaken door een al spelende noot uit te schakelen voordat we de nieuwe afspelen. Dat wordt nog ingewikkelder gemaakt door het feit dat we niet zomaar elke noot willen uitschakelen, maar de noot die al het langst aan het spelen is. Bovendien willen we een grafische interface voor de parameters waarmee we de SID-chip aansturen.

12 Midi-keyboard

Omdat de code van de midi-speler nogal lang is, hebben we geen ruimte om ze hier af te drukken, maar je vindt ze in onze GitHub-repository. **Afbeelding 4** toont de grafische interface. Elke parameter kun je met een muisklik aansturen. De ADSR-parameters kunnen niet continu variëren, maar in vaste stappen met een ongelijke grootte. Met de knopjes + en − stap je daardoor. Het filterblok kun je in- en uitschakelen en de soort filter kun je veranderen. Merk op: er moet minstens één soort filter geselecteerd zijn om geluid te horen. En als je midi-keyboard besturingselementen ingebouwd heeft, kun je ze ook programmeren om deze parameters te veranderen.

En verder...

We kunnen nog heel wat meer mogelijkheden van de SID-chip en geluid uit het Propeller-bordje verkennen. Maar in het laatste deel van deze reeks gooien we het over een andere boeg en bekijken we hoe we met ons bordje VGA-uitvoer genereren.

Deel 03

Mikes Pi-bakkerij: VGA-scherm

Breid je Pi uit met een VGA-uitgang en een retro 8-bits geluidsemulator. Het hart van dit knutselwerk is de achtkerns Propeller-chip. In dit laatste deel genereren we video.



MAKER
Mike Cook

Een oudgediende tijdschriftauteur en schrijver van de serie Body Build. Coauteur van *Raspberry Pi for Dummies*, *Raspberry Pi Projects* en *Raspberry Pi Projects for Dummies*.

magpi.cc/TPaUfT

In het vorige deel van deze reeks maakten we geluid met ons Propeller-ontwikkelbordje; in dit laatste deel gaan we VGA-video genereren. De meeste voorbeelden die je met deze chip ziet, werken zelfstandig. Maar wij willen het met onze Raspberry Pi doen, en dus dienen we de gebaande paden te verlaten. In plaats van met een oud PS/2-toetsenbord en -muis te werken, willen we een seriële verbinding tussen de Pi en Propeller leggen, zoals we in het vorige artikel deden. De beste manier om dat te doen, is iets te veranderen wat al bestaat. Maar dat brengt ons bij enkele interessante problemen die we eerst moeten oplossen.

01 Het VGA-signaal

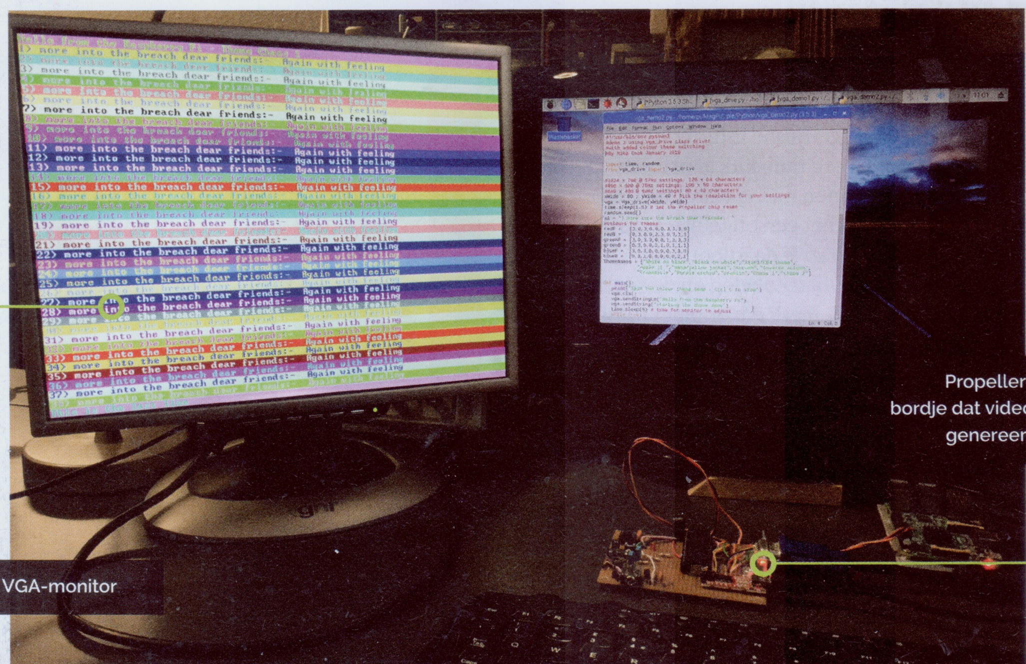
Om een VGA-monitor aan te sturen, dienen we vijf signalen te verwerken: twee synchronisatiepulsen en drie analoge videosignalen. **Afbeelding 1** toont de verbinding tussen de Propeller-chip en de VGA-aansluiting, samen met de acht gebruikte digitale signalen. HS (horizontale synchronisatiepuls) vertelt de monitor wanneer hij van links naar rechts kan beginnen scannen en VS (verticale synchronisatiepuls) wanneer hij van boven naar beneden kan beginnen scannen. De video wordt voor elke kleurcomponent (rood, groen en blauw) gegenereerd met twee digitale lijnen en weerstanden in een eenvoudige dac (digitaal-analoogomzetter). Dat geeft vier mogelijke intensiteiten voor elke kleurcomponent.

02 Het videosignaal

In de VGA-monitor bevindt zich een afsluitweerstand van 75 Ω. Samen met de serieweerstanden vormt dit een spanningsdeler die het signaal van 5 V van de Propeller-chip omlaag haalt. **Afbeelding 2** toont hoe de spanningen die de schakelingen van de VGA-monitor bereiken bepaald

Je hebt nodig

- Propeller-ontwikkelbordje uit MagPi 7 **www.magpi.nl/magazine/2019/7**
- VGA-monitor

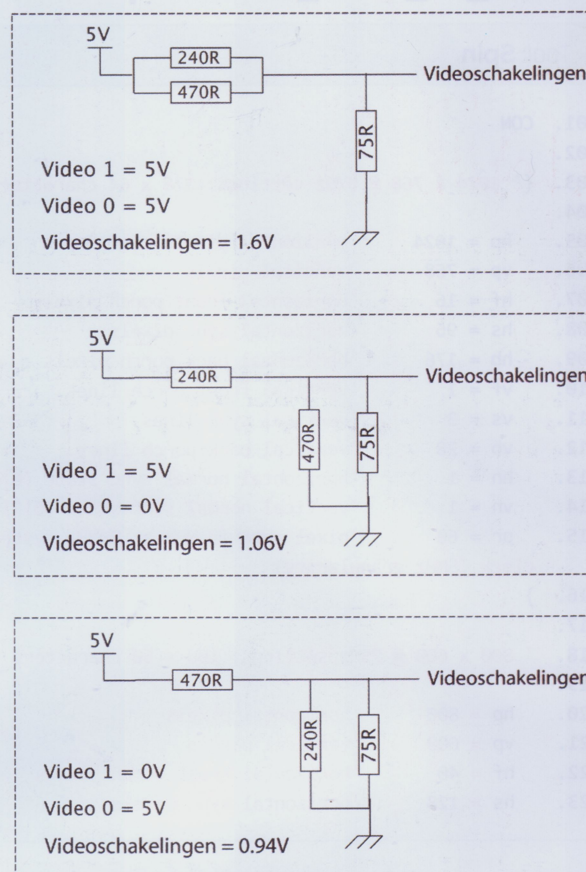


VGA-monitor

Propeller-bordje dat video genereert

Afbeelding 2

Video 1
Video 0
240R
470R
VGA-monitor
Videoschakelingen
75R
Sync
240R
Synchronisatie-schakelingen = 1.2V
75R



worden door welke weerstanden met een logische 1 of 0 verbonden zijn. Deze waardes berekenen we met de formules voor een parallelschakeling en een spanningsdeler, maar uiteindelijk zijn die afgeleid van de wet van Ohm. Merk op: de waardes zijn niet gelijk verdeeld maar hebben een gammacorrectie ondergaan, zodat onze ogen de stappen van helderheid als gelijk beschouwen.

03 Aan de slag

Het officiële Propeller-pakket heet AN004-GUI-StartVGA-Code-v1.0 door André LaMothe. Het is nu best moeilijk om die code te vinden, dus daarom hebben we ze in onze GitHub-repository geplaatst. De code **WMF_HelloWorld_010.spin** kun je rechtstreeks compileren en in het RAM laden. Het is niet al te spannend — het laat een eenvoudig bericht continu over het scherm rollen — maar het is een goede test om te controleren of de verbinding met je VGA-monitor werkt. Als het niet werkt, probeer dan eens de VGA-resolutie naar een andere waarde te veranderen.

P8X32A (DIP)
VGA-socket
5V
(P23) 28 — 240R — R — 1
(P22) 27 — 470R
(P21) 26 — 240R — G — 2
(P20) 25 — 470R
(P19) 24 — 240R — B — 3
(P18) 23 — 470R
(P17) 22 — 240R — HS — 13
(P16) 21 — 240R — VS — 14
9
10
8
7
6
5

Afbeelding 1

04 Verander de videoresolutie

In het begin van de code **VGA_HiRes_Text_10.spin** vind je drie tabellen die de resolutie van het scherm instellen. Standaard staat die op 800×600 @ 75 Hz en toont 100×50 tekens. Gebruik de accolades om een sectie van de code uit te schakelen en verwijder de accolades rond de resolutie die je wilt gebruiken. Deze definities vind je in **Mods_in_VGA_HiRes.spin**. Misschien

▲ **Afbeelding 2** Zet digitale signalen om naar VGA-spanningen

◄ **Afbeelding 1** Schema voor de VGA-uitgang van het Propeller-bordje

Mods_in_VGA_HiRes.spin

> Taal: **Spin**

```
CON

{' 1024 x 768 @ 57Hz settings: 128 x 64 characters

  hp = 1024       'horizontal pixels
  vp = 768        'vertical pixels
  hf = 16         'horizontal front porch pixels
  hs = 96         'horizontal sync pixels
  hb = 176        'horizontal back porch pixels
  vf = 1          'vertical front porch lines
  vs = 3          'vertical sync lines
  vb = 28         'vertical back porch lines
  hn = 1          'horizontal normal sync state (0|1)
  vn = 1          'vertical normal sync state (0|1)
  pr = 60         'pixel rate in MHz at 80MHz system clock (5MHz granularity)
}

' 800 x 600 @ 75Hz settings: 100 x 50 characters

  hp = 800        'horizontal pixels
  vp = 600        'vertical pixels
  hf = 40         'horizontal front porch pixels
  hs = 128        'horizontal sync pixels
  hb = 88         'horizontal back porch pixels
  vf = 1          'vertical front porch lines
  vs = 4          'vertical sync lines
  vb = 23         'vertical back porch lines
  hn = 0          'horizontal normal sync state (0|1)
  vn = 0          'vertical normal sync state (0|1)
  pr = 50         'pixel rate in MHz at 80MHz system clock (5MHz granularity)

' 640 x 480 @ 69Hz settings: 80 x 40 characters
{
  hp = 640        'horizontal pixels
  vp = 480        'vertical pixels
  hf = 24         'horizontal front porch pixels
  hs = 40         'horizontal sync pixels
  hb = 128        'horizontal back porch pixels
  vf = 9          'vertical front porch lines
  vs = 3          'vertical sync lines
  vb = 28         'vertical back porch lines
  hn = 1          'horizontal normal sync state (0|1)
  vn = 1          'vertical normal sync state (0|1)
  pr = 30         'pixel rate in MHz at 80MHz system clock (5MHz granularity)
}
```

reageert je monitor beter op een snellere of tragere kloksnelheid, of misschien wil je gewoon meer informatie op het scherm tonen. Vergeet niet terug te keren naar het tabblad HelloWorld om het bestand in het Propeller-bordje te laden nadat je het veranderd hebt.

Tip



Opstarttijd

Elke keer dat je de seriële poort opent, reset de Propeller-processor zich en laadt hij de VGA-code in. Dat duurt een tijdje, dus we moeten een kleine vertraging toevoegen zodat de VGA-signalen de kans krijgen om op te starten.

05 De juiste bestanden

Creëer een map voor je VGA-generator en kopieer daarin de bestanden **VGA_HiRes_Text_010.spin** en **WMF_Terminal_Services_010.spin** van je download. Voeg ook het bestand **FullDuplexSerialPlus.spi**n toe dat je in onze GitHub-repository vindt of in **magpi.cc/MAYjtg**. Creëer tot slot een map **Python** voor de code die op de Raspberry Pi komt te draaien. Open in de Propeller IDE het bestand **WMF_Terminal_Services_010.spin** en wijzig de functie **OutTerm** in degene die je in **Change_to_Terminal_Services.spin** (drie pagina's verder) vindt. Creëer dan een nieuw bestand en voer de code uit **HelloPi.spin** (de pagina hiernaast) in. Download de code in het RAM van de Propeller, waarna je een bericht op het scherm hoort te zien.

06 Waarom deze veranderingen?

We gaan met het scherm communiceren over een seriële poort, met het datatype string, en daarvoor hebben we een getal nodig om aan te geven dat we aan het einde van de string zijn. Normaal is dat het teken *carriage return*, getal 13 in de ASCII-tekenreeks. De Propeller werkt met strings die je met *null* beëindigt, dus we kunnen ook geen nul als deel van een string sturen. We hebben daarom een strategie nodig die beide getallen vermijdt maar ze nog altijd op het scherm kan tonen. We hebben er daarom voor gekozen om deze besturingscodes andere waardes te geven en een extra getal op te tellen bij de waardes voor de positionering.

07 Kleureninformatie

Kleurenwaardes dienen we op een gelijkaardige manier te manipuleren: zij zijn immers zes bytes lang en kunnen wel eens verkeerd opgevat worden als besturingscodes. Daarom stellen we bit 6 in elke kleurwaarde in en verwijderen dit bit weer wanneer we het

HelloPi.spin

> Taal: **Spin**

```
'' ============================================================================
''   File: HelloPi.Spin
''   Mike Cook
'' ============================================================================
CON
' CONSTANTS, DEFINES, MACROS, ETC.
  _clkmode = xtal1 + pll16x
  _xinfreq = 5_000_000
  RxPin = 31      'default boot loader RX
  TxPin = 30      'default boot loader TX
  BaudRate = 115200 'this is a good baud rate for 80MHz

  ' import some constants from the Propeller Window Manager
  VGACOLS = WMF#VGACOLS
  VGAROWS = WMF#VGAROWS

  ' set these constants based on the Propeller device you are using
  VGA_BASE_PIN         = 16          'VGA pins 16-23

OBJ
  ' Propeller Windows GUI object(s)
  SerIO : "FullDuplexSerialPlus" 'object that implements serial I/O for the Propeller.
  WMF   : "WMF_Terminal_Services_010" ' include the terminal services driver which includes the VGA driver itself

VAR
' DECLARED VARIABLES, ARRAYS

  byte  gVgaRows, gVgaCols ' convenient globals to store number of columns and rows
  byte  gStrBuff1[64]    ' a string buffers
  byte  gTextCursX, gTextCursY, gTextCursMode ' text cursor 0 [x0,y0,mode0]
  long  gVideoBufferPtr ' holds the address of the video buffer passed back from the VGA driver

CON
' MAIN ENTRY POINT
PUB Start | S1, N1, CF, CB

  ' create the GUI itself
  CreateAppGUI
  SerIO.Start(RxPin, TxPin, %0000, BaudRate) ' initialise SerIO object

  ' MAIN EVENT LOOP
  WMF.StringTerm(string("Hello World! from the Propeller "))
  repeat
      S1 := SerIO.getstr(gStrBuff1)
      N1 := byte[gStrBuff1][0]
      if N1 == $0C
        CF := byte[gStrBuff1][1] & $3F
        CB := byte[gStrBuff1][2] & $3F
        WMF.SetLineColor(WMF.GetRowTerm , CF, CB )
      else
        WMF.StringTerm(S1)
      SerIO.tx (2) ' send an ACK to say that has beed processed

' end PUB ------------------------------------------------------------------------

PUB CreateAppGUI | retVal
' This functions creates the entire user interface for the application
  ' start the VGA driver and terminal services
  retVal := WMF.Init(VGA_BASE_PIN, @gTextCursX )

  ' VGA buffer encoded in upper 16-bits of return value
  gVideoBufferPtr := retVal >> 16

  'setup screen colors - see Terminal Services themes
  WMF.ClearScreen( WMF#CTHEME_ATARI_C64_FG, WMF#CTHEME_ATARI_C64_BG )
  'WMF.ClearScreen( WMF#CTHEME_AUTUMN_FG, WMF#CTHEME_AUTUMN_BG )
  return
```

vga_drive.py

> Taal: **Python**

```
#!/usr/bin/env python3
# Class for driving Propeller VDU
# By Mike Cook Jan 2019

import serial

class Vga_drive():

    def __init__(self, chrX,chrY): # initialisation
        self.spinProcessor = serial.Serial( "/dev/ttyUSB0",115200, timeout = 1)
        self.spinProcessor.flushInput()
        self.spinProcessor.flushOutput()
        self.xWidth = chrX -1 # make first line zero
        self.yWidth = chrY -1

    def sendString(self, send): # send a string to the VDU
        buf = 27 # maximum string size for Pi buffers
        self.spinProcessor.flushInput()
        ln = len(send)
        i = 0
        while i < ln:
            self.spinProcessor.write( send[i:i+buf].encode()+ b'\x0D')
            self.waitAck()
            i += buf

    def sendStringLn(self, send): # send a string no CR
        self.sendString(send)
        self.sendNl()

    def sendNl(self): # send a new line command
        self.spinProcessor.flushInput()
        self.spinProcessor.write(b'\x0E\x0D') # send new line
        self.waitAck()

    def setX(self,pos): # set X cursor
        pos += 14 # can't send zero or 13 so add 14
        if pos < 14 or pos > self.xWidth + 14: # outside limits
            print("ERROR ",pos,"IS OUT OF RANGE")
            return
        self.spinProcessor.flushInput()
        s = b'\x0A' + str(chr(pos)).encode() + b'\x0D'
        self.spinProcessor.write(s)
        self.waitAck()

    def setY(self,pos): # set Y cursor
        pos += 14 # can't send zero or 13 so add 14
        if pos < 14 or pos > self.yWidth + 14: # outside limits
            print("ERROR ",pos,"IS OUT OF RANGE")
            return
        self.spinProcessor.flushInput()
        s = b'\x0B' + str(chr(pos)).encode() + b'\x0D'
        self.spinProcessor.write(s)
        self.waitAck()

    def erase(self,line,start,length): # erase "line" from "start" for "length"
        if start + length > self.xWidth:
            length = self.xWidth - start
        s = ""
        for i in range(0,length):
            s +=" "
        if line > -1: # set line to -1 to use current line
            self.setY(line)
        self.setX(start)
        self.sendString(s)
        self.setX(start)
        if line > -1:
            self.setY(line)

    def waitAck(self): #wait for acknowledgement
        ack = self.spinProcessor.read()

    # set colour for current line
    def setColour(self,rf,gf,bf,rb,gb,bb): # r,g,b foreground & r,g,b background
        colF = ((rf & 0x3) << 4) | ((gf & 0x3) << 2) | (bf & 0x3) | 0x40
        colB = ((rb & 0x3) << 4) | ((gb & 0x3) << 2) | (bb & 0x3) | 0x40
        s = b'\x0C' + str(chr(colF)).encode() +str(chr(colB)).encode()+ b'\x0D'
        self.spinProcessor.write(s)
        self.waitAck()

    def cls(self): # clear screen and set to home
        self.spinProcessor.write(b'\x01\x0D') # clear screen
        self.waitAck()

    def home(self): # go to top left
        self.spinProcessor.write(b'\x02\x0D') # home
        self.waitAck()
```

Change_to_Terminal_Services.spin

> Taal: **Spin**

```
PUB OutTerm( pChar )
{{
DESCRIPTION: Output a character to terminal, this is
the primary interface from the client to the driver
in "terminal mode"
direct mode access uses the register model and
controls the engine from the low level

PARMS: pChar - character to print with the following
extra controls:


    $00 = Null - can't be represented by a string
    $01 = clear screen
    $02 = home
    $08 = backspace
    $09 = tab (8 spaces per)
    $0A = set X position (X + 14 follows)
    $0B = set Y position (Y + 14 follows)
    $0C = set color (color follows)
    $0D = return - but can't be passed in a string
from serial
    $0E = return
    others = prints to the screen

    +128 to any other printable character, draw in
inverse video

RETURNS: Nothing.
}}

  case gTermFlag
    $00: case pChar
          $00..$01: bytefill( gVideoBufferPtr,
 ASCII_SPACE, gTermNumCols * gTermNumRows)
                gTermCol := gTermRow := 0

          $02: gTermCol := gTermRow := 0


          $08: if gTermCol
                gTermCol--

          $09: repeat
                PrintTerm(" ")
              while gTermCol & 7

          $0A..$0C: gTermFlag := pChar
              return

          '$0D: NewlineTerm

          $0E: NewlineTerm

          other: PrintTerm( pChar )

    $0A: gTermCol := (pChar-14) // gTermNumCols
    $0B: gTermRow := (pChar-14)// gTermNumRows
    $0C: gTermFlag := pChar & 7

  gTermFlag := 0

' end PUB ---------------------------------------------------
```

ontvangen — zie **afbeelding 3** op het einde van dit artikel. Om er zeker van te zijn dat we geen normale gegevens als kleurwaardes interpreteren, laten we kleurwaardes altijd voorafgaan door het getal 12. En als we de gegevens te snel sturen, lopen de buffers vol — maar met een vertraging werkt het scherm zichzelf langzamer bij. Om dit probleem op te lossen, laten we het scherm een bevestiging terugsturen naar de Pi wanneer het klaar is voor nieuwe gegevens.

08 Met het scherm praten

We willen nu met het scherm communiceren in Python. Om dat eenvoudiger te maken, hebben we een helperklasse geschreven met functies om het scherm in elk Python-programma aan te sturen. Je ziet onze implementatie in het bestand **vga_drive.py** op de vorige pagina. Pyserial zendt en ontvangt alleen byte arrays en deze klasse handelt de omzettingen van en naar strings voor je af. De methode `sendString` werkt met strings van elke lengte: het splitst lange strings op in stukjes van 27 bytes of korter en zendt die elk afzonderlijk. Bekijk elk van de methodes eens om te begrijpen wat ze doen: dit is immers je gereedschapskist.

09 Open je gereedschapskist

Het programma **vga_demo1.py** (op de volgende pagina) toont een eenvoudig programma om getallen en strings naar het scherm te sturen. Eerst maakt dit het scherm leeg, waarna we een bericht naar de bovenste regel sturen. Merk op hoe we een variabele als een string sturen met de functie `str()` en de methode `erase()` gebruiken

Tip



Kleuren

Zodra je een kleurenschema hebt gevonden dat je leuk lijkt, kun je onderaan de HelloPi-code beginnen hacken en de kleuren permanent instellen. Bekijk daarvoor eens de namen van de variabelen in het bestand **WMF_Terminal_Services**.

vga_demo1.py

> Taal: **Python**

```
#!/usr/bin/env python3
#demo 1 using Vga_drive class driver
#By Mike Cook January 2019

import time

from vga_drive import Vga_drive

#1024 x 768 @ 57Hz settings: 128 x 64 characters
#800 x 600 @ 75Hz settings: 100 x 50 characters
#640 x 480 @ 69Hz settings: 80 x 40 characters
xWide = 100 ; yWide = 50 # Pick the resolution for your VDU settings
vga = Vga_drive(xWide, yWide)
time.sleep(1.5) # let the Propeller chip reset
s1 = ") Now is the time to come to the aide of the party "
s2 = " Again with feeling "

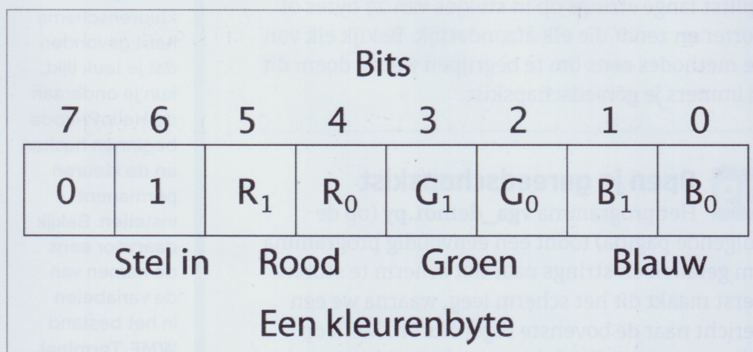
def main():
    print("Spin VGA driver demo1 - Ctrl C to stop")
    vga.cls()
    speed = 0.05 # time between each change
    while True:
        vga.sendStringLn(" Hello from the Raspberry Pi") # send with CR
        for i in range(1,yWide+20): # allow scrolling at the end
            vga.sendString(str(i)+s1) # send number and string
            time.sleep(speed*4)
            vga.sendString(s2)
            time.sleep(speed * 8)
            vga.erase(-1, len(s1)+len(str(i)), len(s1)) # remove last string
            time.sleep(speed)
            vga.sendNl() # new line
        vga.cls()

# Main program logic:
if __name__ == '__main__':
    main()
```

om een deel van een regel met spaties te vullen. Als we het regelnummer -1 gebruiken, werkt het op de huidige regel, maar je kunt ook een specifiek regelnummer gebruiken. Het programma kan meer regels schrijven dan er op het scherm passen, dus je kunt het scherm dan zien scrollen.

10 Kleurenschema's

In de VGA-drivers is er slechts één bit geheugen per pixel. Maar dit bit kun je weergeven als twee van de 64 beschikbare kleuren. Die kleuren kun je instellen, maar slechts regel per regel. De voorgrond- en achtergrondkleuren kun je onafhankelijk van elkaar instellen, maar ze moeten uiteraard verschillend zijn als je iets wilt zien. Het programma **vga_demo2.py** van de volgende pagina toont tien verschillende kleurenschema's die we aanraden, evenals twee leuke chaotische of willekeurige schema's. Merk op: we kunnen de bovenste regel omgekeerd maken van de rest van het scherm.



Afbeelding 3 De samenstelling van een kleurenbyte

11 Conclusie

Daarmee sluiten we dit project na drie artikels af, maar er valt nog heel wat te verkennen. Bekijk de voorbeelden eens die meerdere bits per pixel hebben, waarmee je meer kleuren en meer grafische mogelijkheden verkrijgt. In onze GitHub-repository vind je een voorbeeld. We hebben ook niet alle gpio-pennen van de processor naar buiten geleid. Met een headerstrip kun je eenvoudig de massa en de eerste acht pennen bereiken. Alle andere ongebruikte pennen liggen ook nog voor het oprapen. Hopelijk hebben we je duidelijk gemaakt dat de Propeller-processor een van de interessantste en veelzijdigste componenten is die je op je Raspberry kunt aansluiten.

vga_demo2.py

> Taal: **Python**

DOWNLOAD DE VOLLEDIGE CODE:


magpi.cc/XCwzVh

```python
#!/usr/bin/env python3
#demo 2 using Vga_drive class driver
#with added colour theme switching
#By Mike Cook January 2019

import time, random
from vga_drive import Vga_drive

#1024 x 768 @ 57Hz settings: 128 x 64 characters
#800 x 600 @ 75Hz settings: 100 x 50 characters
#640 x 480 @ 69Hz settings: 80 x 40 characters
xWide = 100 ; yWide = 50 # Pick the resolution for your settings
vga = Vga_drive(xWide, yWide)
time.sleep(1.5) # let the Propeller chip reset
random.seed()
s1 = ") more into the breach dear friends:- "
#colours for themes
redF =   [3,0,3,0,0,0,3,3,3,0]
redB =   [0,3,0,0,2,3,0,3,1,1]
greenF = [3,0,3,3,0,0,1,3,3,3]
greenB = [0,3,0,0,2,1,0,1,1,1]
blueF =  [3,0,3,0,0,0,0,3,3,0]
blueB =  [0,3,2,0,0,0,0,0,2,1]
ThemeNames = ["White on black","Black on white","Atari/C64 theme",
              "Apple ][ ","Wasp/yellow jacket","Autumn","Inverse Autumn",
              "Creamsicle","Purple orchid","Gremlin","Chaos 1","Chaos 2"]

def main():
    print("Spin VGA colour theme demo - Ctrl C to stop")
    vga.cls()
    vga.sendStringLn("Hello from the Raspberry Pi")
    vga.sendString("starting the theme demo")
    time.sleep(5) # time for monitor to adjust
    while True:
        for j in range(0,12):
            if j >= 10:
                chaos(j & 1)
            else :
                changeTheme(j,1) # use 0 for no special top line
            vga.sendStringLn(
            "Hello from the Raspberry Pi - Theme "+ThemeNames[j])
            for i in range(1,yWide-1):
                vga.sendString(str(i)+s1) # send number and string
                vga.sendStringLn(" Again with feeling ")
            vga.sendString("This is the last line")
            time.sleep(2)
            vga.cls()

def changeTheme(theme, top): # top = 1 if inverted
    start = top & 1
    if start != 0:
        vga.setY(0)
        vga.setColour(redB[theme],greenB[theme],blueB[theme],
                      redF[theme],greenF[theme],blueF[theme])

    for i in range(start, yWide):
        vga.setY(i) # choose line
        vga.setColour(redF[theme],greenF[theme],blueF[theme],
                      redB[theme],greenB[theme],blueB[theme])
    vga.home()

def chaos(t): #just a bit of fun
    for i in range(0,yWide):
        vga.setY(i) # choose line
        c = [random.randint(0,3) for j in range(0,6)]
        if t == 1 :
            vga.setColour(c[0],c[1],c[2],c[3],c[4],c[5])
        else:
            vga.setColour(c[0],c[1],c[2],c[0]^3,c[1]^3,c[2]^3) # inverse back
    vga.home()


# Main program logic:
if __name__ == '__main__':
    main()
```



Eenvoudig compileren

AUTHEUR
Richard Smedley

Omdat Richard woorden altijd al beter vond dan naar dingen wijzen, hield hij vast aan zijn opdrachtregel waarvan alle anderen weggevlucht waren.

@RichardSmedley

Software vanuit de broncode bouwen en installeren hoeft niet moeilijk te zijn.

De meeste software is heel eenvoudig in Raspbian te installeren — tenminste als ze volwassen genoeg is om als een deb-archiefbestand verpakt te zijn. Maar vaak vind je geweldige software die je graag eens zou uitproberen, maar waarvoor geen deb-bestand bestaat. Je krijgt dan de vraag om ze te installeren — soms na ze eerst van GitHub te 'klonen'.

Hoewel het niet zo eenvoudig is als een apt-get install opdracht uitvoeren, hoef je niet bang te zijn om door het tientallen jaren oude ritueel van software compileren te gaan. En wanneer je toch tegen een foutmelding aanloopt, is het vaak vrij eenvoudig om weer op het goede spoor te komen. We kijken ook naar scripts om Python-pakketten te installeren, maar laten we eerst eens uitzoeken hoe we een pakket in het verkeerde formaat kunnen installeren.

Raspbian is niet de enige GNU/Linux-distributie die op Debian gebaseerd is — distrowatch.com toont er meer dan honderd. Je zult waarschijnlijk ook al gehoord hebben van Ubuntu en Linux Mint. De naam Ubuntu is afgeleid van een woord uit een Nguni Bantu-taal met de betekenis "menselijk zijn voor anderen". De distributie is zo populair dat veel projecten, inclusief degene die niet in de repository's van Debian en Raspbian zitten, deb-pakketten voor elke Ubuntu-versie uitbrengen.

Ubuntu introduceerde ook het concept van Personal Package Archives (PPA) in de Debian-familie. Dit zijn speciale softwarerepository's waarop ontwikkelaars bronpakketten kunnen publiceren zoals op de officiële Ubuntu-repository, maar dan voor onofficiële software. Ppa's zul je niet snel vinden voor Pi-software, maar als je interesse hebt gekregen om Raspbians neefje Ubuntu op je pc uit te proberen, bekijk dan zeker de communitydocumentatie van Ubuntu.

Houd je installaties bij

Als je software van buiten Raspbians repository installeert, moet je er afzonderlijk naar omkijken. Dat betekent dat je zelf updates dient te installeren met oplossingen voor fouten en beveiligingsproblemen. Volg daarom goed op wat je installeert.

Beschikbare pakketten

Als je `apt-get` uitvoert of met `apt-cache search` naar een pakket zoekt, gaat apt eerst in zijn lokale pakketlijst zoeken welke pakketten er beschikbaar zijn. De lijst met bronnen waar apt deze pakketten haalt — de url's van de repository's — houdt het programma bij in het bestand **/etc/apt/sources.list** en bestanden in de directory **/etc/apt/sources.list.d/**. Aan die directory kun je ook manueel een bestand met de url van een repository toevoegen als een project waarin je geïnteresseerd bent er een bijhoudt.

Je kunt ook **/etc/apt/sources.list** aanpassen, maar dat raden we niet aan. Soms komt het van pas: zo was **wheezy** naar **jessie** veranderen in dit bestand een (niet ondersteunde) manier om naar Jessie te upgraden zonder je sd-kaart helemaal opnieuw te moeten herschrijven toen de Raspbian Jessie upgrade uitkwam.

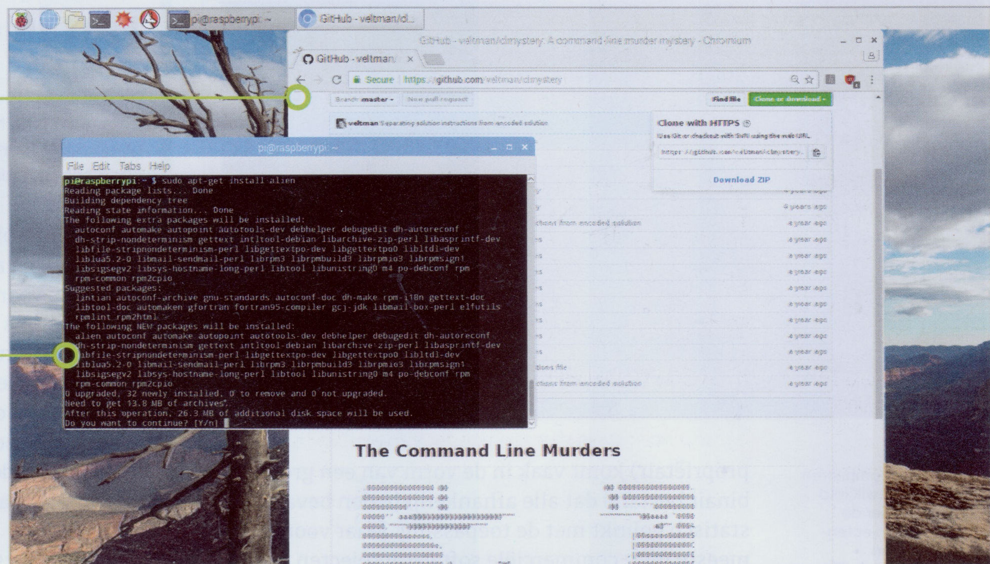
Je kunt ook gewoon een deb-bestand downloaden en dan met de volgende opdracht installeren:

```
sudo dpkg -i eenpakket.deb
```

En als het pakket van andere pakketten afhangt, installeer je die daarna:

```
sudo apt-get install -f
```

Naast de Debian-familie is er nog een andere populaire familie van GNU/Linux-distributies, gebaseerd op Red Hat, inclusief Fedora en CentOS. Fedora is voor de Pi 2 of 3 beschikbaar als alternatief voor Raspbian, als je het wilt uitproberen. Distributies uit de Red Hat-familie verspreiden hun pakketten als rpm-archiefbestand (Red Hat Package Management). Als je een pakket dat voor één distributiefamilie



Typ `git clone` in en plak daarna de url die eindigt op .git om de nieuwste broncode te downloaden.

Na de installatie van alien kun je pakketten omzetten tussen de formaten van Raspbian, Fedora en andere systemen.

gemaakt is op een distributie van een andere familie wilt installeren, komt het pakket alien ter hulp, dat pakketten van rpm naar deb kan omzetten en andersom.

Zo zal:

```
alien eenpakket.rpm
```

...het rpm-bestand naar deb-formaat omzetten. Ook andersom kan, een deb-bestand naar rpm-formaat omzetten:

```
alien -r eenpakket.deb
```

./configure && make && sudo make install

Voordat pakketbeheerders bestonden, was het normaal om zelf de broncode te downloaden (meestal in de programmeertalen C of C++ geschreven), de code te compileren met de compiler gcc, bibliotheken te linken en het resulterende programma dan op de correcte plaats op je harde schijf te installeren.

Heel wat hindernissen die je toen diende te doorstaan, zijn al lang verleden tijd: `configure` en Makefiles handelen nu al het moeilijke werk af: ze controleren of afhankelijkheden geïnstalleerd zijn, gebruiken de juiste compilervlaggen voor het project en het platform en installeren zelfs de manpagina op de juiste plaats.

Hoewel de meeste software die je wilt draaien als een deb-bestand beschikbaar is om te installeren met `apt` of `dpkg`, of een shellscript meelevert dat de installatie uitvoert (zie verder), dien je heel wat projecten, vooral degene die je op GitHub vindt, zelf uit te pakken en te compileren. Pak het archiefbestand eerst uit met:

```
tar xvf desoftware.tgz
```

Ga daarna in de uitgepakte directory met `cd desoftware` en zoek naar een bestand **README**, **README.md** of misschien **INSTALL**. Dat vertelt je gewoonlijk om drie opdrachten uit te voeren:

```
./configure
make
sudo make install
```

Maar lees zeker de instructies, want er zijn variaties mogelijk, en sommige software omzeilt `make` zelfs met een eigen lokale versie die je dan als `./make` dient uit te voeren.

Als je op de Pi zomaar `./configure` uitvoert, is de kans groot dat programma's (en in het bijzonder bibliotheken) in een niet-standaardlocatie geïnstalleerd worden — **/usr/local/lib** in plaats van **/usr/lib**. Dat kan tot problemen leiden als die directory niet in het zoekpad voor bibliotheken zit. En als je een nieuwe versie van een bibliotheek compileert die al op je systeem staat, eindig je met twee versies van de bibliotheek op je systeem, dat de oude blijft gebruiken omdat het de nieuwe niet vindt. In plaats van gewoon `./configure` uit te voeren, raden we daarom het volgende aan om problemen te vermijden:

```
./configure --prefix=/usr --libdir=/usr/lib/arm-linux-gnueabihf
```

Afhankelijkheden

Commerciële software (zowel opensource als

```
pi@raspberr... ✕ | pi@raspberr... ✕
        + ffmpeg > Not Available
          > movie2gif won't work

< Optional SuperUser Applications
        + dmidecode > OK

< Required Build Tools
        + msgfmt > OK

< Python Interpreter > 2.7.0 && < 3.9.0
        + Python > OK

< Required Python Modules
        + gettext > OK
        + Gtk (gi.repository) > OK
        + configobj > OK
        + shutil > OK

Notes from configure:
Prefix                  > /usr
Python                  > /usr/bin/python3
Post-Install Tasks      > Enabled

        You may want to continue with './make build'.

pi@raspberrypi:~/Downloads/bashstyle-ng $
```

▲ Software compileren kan er ingewikkeld uitzien, maar de meeste projecten leveren een degelijke **Makefile** mee om het proces te stroomlijnen. Sommige tonen zelfs hints.

propriëtair) komt vaak in de vorm van een groot binair bestand dat alle afhankelijkheden bevat, statisch gelinkt met de toepassing. Maar voor de meeste niet-commerciële softwareprojecten is het de gewoonte om eenvoudigweg een lijst mee te leveren van welke versies van welke bibliotheken nodig zijn om de software te compileren en uit te voeren. Gelukkig is er een eenvoudige opdracht om alle afhankelijkheden te installeren die je nodig hebt om een Debian-pakket te compileren:

```
sudo apt-get build-dep <pakketnaam>
```

Af en toe dien je eerst iets anders te compileren en te installeren. Vaak is de reden dat je zo de nieuwste versie van de software verkrijgt, met nieuwe functies, een betere compatibiliteit of oplossingen voor fouten. Tegenwoordig is GitHub

> "GitHub is de standaardlocatie geworden om vrije software te verspreiden"

de standaardlocatie geworden om vrije software te verspreiden. Er bestaan wel andere repository's, maar we kijken in dit artikel naar hoe je software van GitHub downloadt.

Je eerste kennismaking met GitHub is misschien dat je op een projectpagina een knopje ziet dat je uitnodigt met "fork me on GitHub" of dat je vraagt om de software te "klonen". Ja dit project leeft in een wereld die groot genoeg is om zijn eigen jargon te hebben. Een "fork" is eenvoudigweg je eigen kopie als je mee aan het project helpt ontwikkelen, waarna je je wijzigingen weer aan het project kunt aanbieden of ze (op GitHub of elders) kunt publiceren zodat anderen het kunnen uitproberen of erop kunnen voortbouwen.

Je hoeft niet eens te kunnen programmeren. Heel wat mensen gebruiken GitHub om samen te werken aan documentatie, wetenschappelijk onderzoek of zelfs fictie. Maar om eenvoudigweg iets van GitHub te downloaden, hoef je maar één ding te weten — hoe je de broncode van de toepassing kunt "klonen" naar je Pi. Dat gaat met de volgende opdracht:

```
git clone https://github.com/veltman/
clmystery.git
```

Deze opdracht maakt een lokale directory aan met alle bronbestanden. Ga met `cd clmystery` naar de directory die je juist aangemaakt hebt en volg de instructies in **README.md**.

Het is een mysterie

De software **clmystery** in ons voorbeeld is The Command Line Mystery — een tekstgebaseerd game dat je leert om de opdrachtregel te gebruiken. Amusant en heel nuttig!

Nil desperandum

Soms gaat er iets mis tijdens het compileren, en dan breekt het script bijvoorbeeld af met een klacht over een ontbrekend pakket. Op een goede dag voer je een `apt-cache search` uit voor de naam van de ontbrekende afhankelijkheid en vind je het pakket, dat je dan eenvoudig op Raspbian kunt installeren.

Op een iets minder goede dag dien je wat rond te graven om (de nieuwste versie van) de software te vinden die je nodig hebt. Dat vereist misschien een trip naar GitHub of SourceForge. Dat is OK als de software-omgeving van de ontwikkelaar op die van jou lijkt: de extra installatiestappen kunnen dan vlot verlopen. In het andere geval dien je niet te wanhopen: er is hulp beschikbaar.

De meeste projecten hebben een of meer manieren om een probleem te rapporteren en hulp te zoeken: een mailinglijst of Google-groep, een wiki op de SourceForge-pagina van het project, een e-mailadres van de hoofdontwikkelaar of zelfs een Twitter-account. Probeer je probleem met zoveel mogelijk details uit te leggen en blijf beleefd en geduldig. Meestal zul je dan behulpzame mensen tegenkomen. Denk er altijd aan dat je mensen vraagt om hun tijd op te geven om jou te beantwoorden. Als je gefrustreerd bent door het installatieproces — en dat hebben we allemaal wel eens meegemaakt — blijf dan altijd respectvol tegenover iedereen die de moeite doet om je te helpen.

Als je geen tijd hebt om op antwoorden te jagen maar je interesse blijft hebben in een project,

▼ Git is ontwikkeld door Linux-ontwikkelaar Linus Torvalds om met de complexiteit van meerdere miljoenen regels kernelcode om te kunnen, maar vereenvoudigt zelfs het versiebeheer van de kleinste softwareprojecten.

```
pi@raspberr... ✕ | pi@raspberr... ✕
pi@raspberrypi:~/Downloads $ git clone https://github.com/veltman/clmystery.git
Cloning into 'clmystery'...
remote: Counting objects: 1000, done.
remote: Total 1000 (delta 0), reused 0 (delta 0), pack-reused 1000
Receiving objects: 100% (1000/1000), 6.21 MiB | 6.00 KiB/s, done.
Resolving deltas: 100% (50/50), done.
Checking connectivity... done.
pi@raspberrypi:~/Downloads $
```

```
pi@raspberr...  ✖  pi@raspberr...  ✖
  GNU nano 2.2.6                    File: install.sh

  echo 'Removing old download...'
  rm -f edublocks-$ARCH.tar.xz
fi

if [ -d edublocks ]; then
  echo ''
  echo 'Removing old extract...'
  rm -rf edublocks
fi

echo ''
echo 'Downloading package...'
wget http://edublocks.org/downloads/edublocks-$ARCH.tar.xz

echo ''
echo 'Extracting package...'
tar -xf edublocks-$ARCH.tar.xz

echo ''

^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

EduBlocks is een goed voorbeeld van een project dat je leven eenvoudig maakt door elke fase van de installatie in een shellscript te plaatsen, zodat je slechts één opdracht hoeft uit te voeren om de software te installeren.

is een andere optie om op de volgende versie van het project te wachten. Tegen dan zijn de afhankelijkheden wellicht algemener beschikbaar en hebben ze misschien zelfs hun weg gevonden naar Raspbians repository's. Of misschien komt het probleem dan eenvoudigweg niet meer voor.

Installatie met een script

Soms krijg je de vraag om een installatiescript te downloaden en het rechtstreeks uit te voeren. Dat is bijvoorbeeld het geval met het uitstekende EduBlocks, dat zich half tussen Scratch en Python bevindt en jonge programmeurs helpt om de grote stap tussen de twee talen te zetten. EduBlocks installeren doe je als volgt:

```
curl -sSL get.edublocks.org | bash
```

Daarmee download je het shellscript op **get.edublocks.org** (je kunt het zelfs in je webbrowser bekijken) en stuurt het door een sluisteken (pipe) naar een Bash-proces om uit te voeren. De optie -s vertelt curl om geen voortgangsindicator of foutmeldingen te tonen, -S overschrijft dit zodat je wel een foutmelding krijgt als de download niet lukt en -L vertelt curl dat het een doorverwijzing van de website moet volgen.

Als je er niet van houdt om software op je Pi uit te voeren zonder dat je weet wat die doet, of als je eenvoudigweg een kijkje wilt nemen in het script en wilt zien wat de installatie doet, download het bestand dan en sla het op als install.sh als dat geen bestand met dezelfde naam overschrijft in je huidige directory:

```
curl -o install.sh -L get.edublocks.org
```

Als je dan het script **install.sh** inkijkt, zie je dat het een tarball downloadt en uitpakt met meerdere scripts daarin, en dat het eerst een script uitvoert om de afhankelijkheden te installeren.

Je kunt dit script uitvoeren met `sh install.sh`. Als je het eerst niet ingekeken hebt, besef dan dat je veel vertrouwen hebt in niet alleen de ontwikkelaars die het script schreven, maar in elke tussenstap van zijn reis van de EduBlocks-server tot je machine.

Python-pakketten

Als je al jaren software uit allerlei bronnen installeert, leer je te herkennen wanneer het waarschijnlijk een pijnloze installatie wordt. Eén reden tot hoop is als de software in Python geschreven is. Uiteraard kun je in elke programmeertaal zowel goede als slechte software schrijven, maar Python-software en Python-bibliotheken lijken vaak betrouwbaar te zijn en gemakkelijk als pakket te installeren.

Hoewel Debian-gebaseerde GNU/Linux-distributies zoals Raspbian met het uitstekende apt en dpkg (Debian Package Manager) komen, hebben veel populaire programmeertalen hun eigen ecosystemen van pakketbeheerders en repository's uitgebouwd. Er zijn er zelfs meerdere alleen maar voor de teksteditor Emacs! De twee die je het vaakst zult tegenkomen, zijn JavaScripts npm en Pythons pip. Als je apt gewoon bent, zal de opdracht om een Python-pakket met pip te installeren vertrouwd overkomen:

```
sudo pip3 install numpy
```

Dat is de opdracht die je uitvoert als de installatie-instructies je opdragen om de Python-bibliotheek NumPy te installeren. Ga nu maar aan het installeren! En maak je geen zorgen — het is best moeilijk om je Raspbian-installatie te verknoeien. En als je er toch in slaagt, kun je tenminste nog die back-ups die je in de vorige MagPi gemaakt hebt eens testen.

Installeer eens Ubuntu

Ubuntu kun je installeren op een Pi 2 of 3, maar ook op je pc of Mac.



Maak een machine die je dingen kunt aanleren met Google's **Edge TPU**

Gebruik een Coral USB Accelerator en een Raspberry Pi Camera om een apparaat te bouwen dat je kunt leren om objecten te herkennen. Door **Lucy Hattersley.**

coral.withgoogle.com

MAKER

Mike Tyka

Mike werkt met kunstmatige neurale netwerken als een artistiek medium en hulpmiddel. Hij creëerde enkele van de eerste kunstwerken op grote schaal met Iterative DeepDream. Hij werkte ook samen met Refik Anadol om baanbrekende meeslepende projectie-installaties zoals Archive Dreaming te ontwikkelen met Generative Adversarial Networks. Momenteel werkt Mike aan machinaal leren bij Google in Seattle.

miketyka.com

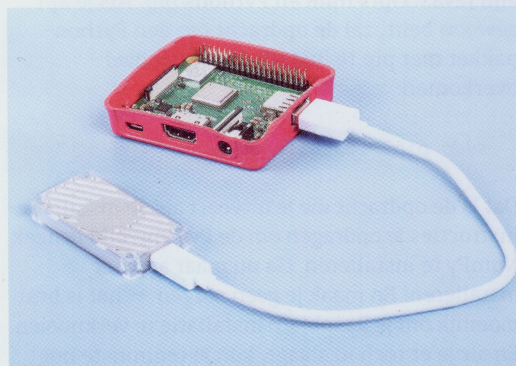
Je hebt nodig

- Raspberry Pi 3B / 3B+
- Coral USB Accelerator (€ 70)
- Raspberry Pi Camera Module
- 8 Gbyte microSD-kaart
- "Raspbian Stretch with desktop" image
- Experimenteerbord
- 5 × Drukknoppen
- 10 × Jumperdraden (mannelijk naar vrouwelijk)
- 9 × Korte jumperdraden (mannelijk naar mannelijk)
- 4 × Weerstanden (330 Ω)
- 4 × leds

De nieuwe Coral USB Accelerator van Google (zie ook pagina 8) helpt je om de mogelijkheden van AI in elk Raspberry Pi-project in te bouwen. Het apparaat is gebouwd rond de Coral Edge TPU-chip, een asic (*application-specific integrated circuit*) die de prestaties van neurale netwerken op een apparaat enorm versnelt.

Als je dit combineert met de Raspberry Pi en de Raspberry Pi Camera, heb je een volledig systeem dat perfect is om complexe computervisietaken uit te voeren, zoals objecten herkennen. Omdat de versneller lokaal werkt, heb je geen verbinding met een clouddienst nodig en hoef je geen gegevens over internet te delen. Daardoor draait het systeem ook met minder vertraging dan een cloudverbinding en kan het bijna realtime objecten herkennen.

Met de Raspberry Pi Camera Module voeg je eenvoudig fotofunctionaliteit toe aan alle Raspberry Pi-bordjes. En met de ingebouwde gpio-pennen kun je prototypes van schakelingen op de Pi aansluiten en de Pi zelfs in andere projecten en



▲ De USB Accelerator verbonden met een Raspberry Pi

industriële omgevingen integreren. Met een USB Accelerator erbij heb je een krachtig apparaat dat allerlei taken op zich kan nemen.

In dit praktische artikel gaan we een machine bouwen die je dingen kunt aanleren. Het project is ontworpen door Mike Tyka van Google.

De machine leert om objecten te identificeren die je ervoor houdt. Die objecten kunnen divers zijn: sleutels, fruit, schaakstukken of zelfs vingers of gezichten.

De gebruiker houdt objecten voor de camera van de Pi en drukt op een knop. Het apparaat onthoudt dit object dan — als het dit object opnieuw ziet, laat het de corresponderende led oplichten.

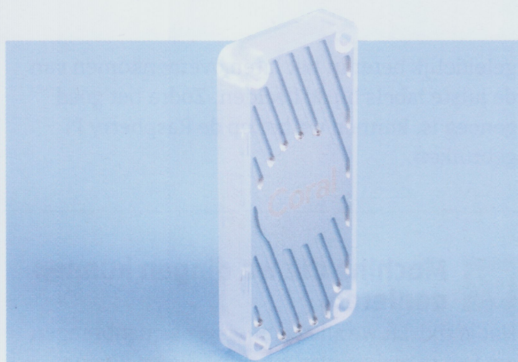
Deze machine is een geweldig voorbeeld van hoe je eenvoudig een laagje machinaal leren aan je projecten kunt toevoegen zonder dat je een heel netwerk vanaf nul dient te trainen.

De machine kan objecten snel en efficiënt onderscheiden — zelfs vanuit verschillende gezichtspunten. Dit soort project was enkele jaren geleden zelfs niet mogelijk op een krachtige computer met een dure grafische kaart. En nu voegen we het aan een klein computerbordje zoals een Raspberry Pi toe!

01 Start met Raspbian

Start met het image "Raspbian Stretch with desktop" op een microSD-kaart te schrijven. Zie daarvoor onze snelstartgids in MagPi 6 (**www.magpi.nl/magazine/2019/6**).

Sluit dan de Raspberry Pi Camera Module met een lintkabel aan op je Pi — zie ons artikel over de cameramodule in MagPi 6. Steek dan de microSD-kaart in je Pi en sluit een voedingskabel aan om de Pi op te starten.



▲ De USB Accelerator stelt een Raspberry Pi in staat om AI-taken realtime uit te voeren.

02 Stel de camera in

Nadat Raspbian opgestart is, klik je op het menu-icoontje van de Raspberry Pi helemaal bovenaan links en kies je **Preferences / Raspberry Pi Configuration**. Klik op **Interfaces**, zet **Camera** op **Enabled** en klik **OK**. Er verschijnt dan een waarschuwing **Reboot needed**. Klik op **Yes**.

Open een Terminal-venster (**Ctrl+Alt+T**), neem een testfoto en open het bestand om het te bekijken:

```
raspistill -v -o test.jpg
xdg-open /home/pi/test.jpg
```

03 Stel de USB Accelerator in

Verzeker je ervan dat de USB Accelerator niet verbonden is terwijl je de software instelt. Open een Terminal-venster en voer de volgende opdrachten in om de software te downloaden en te installeren:

```
wget http://storage.googleapis.com/
cloud-iot-edge-pretrained-models/edgetpu_
api.tar.gz
tar xzf edgetpu_api.tar.gz
cd python-tflite-source
bash ./install.sh
```

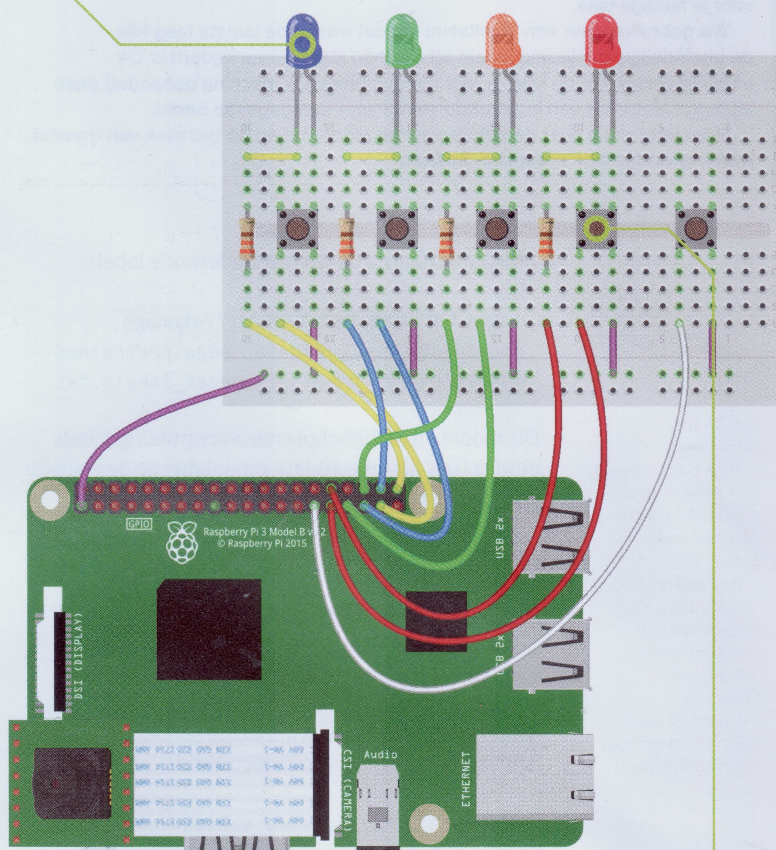
Tijdens de installatie krijg je de vraag **Would you like to enable the maximum operating frequency?**. Antwoord voorlopig n. Je kunt dit altijd later nog inschakelen als je de prestaties wilt verhogen.

04 Test de USB Accelerator

Sluit nu de USB Accelerator op de Pi aan met de meegeleverde kabel die USB type A naar USB type C omzet. Je USB Accelerator is nu klaar.

Voor meer informatie over het instellen en testen kun je terecht in het document 'Get started with the USB Accelerator' van Coral (**g.co/coral/setup**).

Afbeelding 1



05 Download een model

Laten we de meegeleverde democode eens bekijken. Het script **classify_capture.py** werkt met de Raspberry Pi Camera Module om live beeldclassificatie van objecten rond ons uit te voeren.

Daarvoor hebben we een model nodig dat getraind is om enkele alledaagse voorwerpen te herkennen. Download bijvoorbeeld dit MobileNet-model dat getraind is om 1000 voorwerpen te herkennen:

```
wget -P test_data/ https://storage.
googleapis.com/cloud-iot-edge-pretrained-
models/canned_models/mobilenet_v2_1.0_224_
quant_edgetpu.tflite
```

Voorzichtig!

De USB Accelerator kan tijdens het gebruik heet worden. Raak hem niet aan terwijl je project loopt.

Tip



Hoe het werkt: transfer learning

De machine in dit artikel gebruikt een aanpak die transfer learning genoemd wordt. Met deze techniek start je met een al getraind model en pas je het aan voor je huidige taak.

We gebruiken hier een MobileNet-model waarin de laatste laag (die de uiteindelijke beslissing maakt uit de 1000 klassen) verwijderd is. De uitgangsvector van de laag ervoor ligt dus bloot. De machine behandelt deze uitgangsvector als een ingebedde vector voor een gegeven beeld.

Meer informatie over de details van het algoritme en de techniek van transfer learning vind je hier: **magpi.cc/LeKzkc**.

Download daarna de corresponderende labels:

```
wget -P test_data/ http://storage.
googleapis.com/cloud-iot-edge-pretrained-
models/canned_models/imagenet_labels.txt
```

Dit model en de bijbehorende verzameling labels vind je (net als vele andere modellen) op de website van Coral (**coral.withgoogle.com**).

06 Live cameradetectie

We hebben nu een voorgetraind model voor 1000 objecten en een corresponderende verzameling van labels, plus een script dat live de camera opneemt. Neem deze drie samen en je hebt een camera die objecten detecteert:

```
python3 demo/classify_capture.py --model
test_data/mobilenet_v2_1.0_224_quant_
edgetpu.tflite --label test_data/imagenet_
labels.txt
```

Beweeg de Raspberry Pi Camera Module rond. Het voorvertoningsvenster zal de objecten om je heen identificeren: laptop, muis, blikje frisdrank enzovoort.

Het programma **classify_capture.py** gebruikt twee opties:

```
--model
--label
```

Elk van deze opties vul je aan met het model en de corresponderende labels die je in **stap 05** gedownload hebt.

Dit is een TensorFlow Lite-model (vandaar de bestandsextensie .tflite) dat voorgetraind is om 1000 objecten te herkennen. De training is meestal op een veel snellere computer of een clouddienst gebeurd, met duizenden trainings- en testbeelden. Tijdens het trainen wordt het model geleidelijk beter in het laten overeenkomen van de juiste labels bij de beelden. Zodra het goed genoeg is, kunnen we het op de Raspberry Pi gebruiken.

Tip



Modellen verkrijgen

Op de website van Coral stelt Google een selectie van modellen beschikbaar: **magpi.cc/OmyrGC**

07 Machine die we dingen kunnen aanleren

Het is tijd dat we alle technologie samenbrengen om iets te maken: een machine die we dingen kunnen aanleren. Zoals onze demo scant ons project naar objecten met de Raspberry Pi Camera Module. Een verschil is dat onze machine kan leren om de objecten te detecteren die je ervoor houdt.

Start met enkele extra afhankelijkheden te installeren:

```
sudo apt-get install libgstreamer1.0-0
gstreamer1.0-tools gstreamer1.0-
plugins-base gstreamer1.0-plugins-good
gstreamer1.0-plugins-bad gstreamer1.0-
plugins-ugly python3-gst-1.0 python3-gi
```

Voeg daarna **bcm2835-v4l2** toe aan het einde van **/etc/modules**:

```
echo bcm2835-v4l2 | sudo tee -a /etc/
modules
```

08 Bouw de machine

Schakel de Raspberry Pi uit met deze Terminal-opdracht (of kies Menu / Shutdown en klik dan op Shutdown):

```
sudo shutdown -h now
```

Verwijder de voedingskabel van de Pi en zet nu het experimenteerbordje op met de schakelaars en leds zoals je in het diagram van de schakeling ziet (**afbeelding 1**). Leg de Raspberry Pi Camera Module plat neer, zodat hij naar boven gericht is.

Zodra de schakeling klaar is, sluit je de voedingskabel weer aan.

09 Installeer de code

Open de webbrowser Chromium en bezoek onze GitHub-pagina (**magpi.cc/github79**). Download daar het bestand **teachable_rpi3.tgz** naar je persoonlijke map. Open een Terminal-venster en pak de code uit:

```
tar xvzf teachable_rpi3.tgz cd /home/pi/
teachable/
```

embedding.py

DOWNLOAD DE VOLLEDIGE CODE:

 **magpi.cc/github79**

> Taal: **Python**

```python
"""Detection Engine used for detection tasks."""
from collections import Counter
from collections import defaultdict
from edgetpu.basic.basic_engine import BasicEngine
import numpy as np
from PIL import Image


class EmbeddingEngine(BasicEngine):
  """Obtains embedding from headless mobilenets."""

  def __init__(self, model_path):
    """Creates a EmbeddingEngine with given model.
    Args:
      model_path: Path to TF-Lite Flatbuffer file.
    """
    BasicEngine.__init__(self, model_path)
    output_tensors_sizes = self.get_all_output_
tensors_sizes()
    if output_tensors_sizes.size != 1:
      raise ValueError(
          ('Model should have only 1 output tensor'
           'This model has {}.'.format(
               output_tensors_sizes.size)))

  def DetectWithImage(self, img):
    """Calculates embedding from an image.
    Args:
      img: PIL image object.
    Returns:
      Embedding vector as np.float32
    """
    input_tensor_shape = self.get_input_tensor_
shape()
    if (input_tensor_shape.size != 4 or
        input_tensor_shape[3] != 3 or
        input_tensor_shape[0] != 1):
      raise RuntimeError(
          'Invalid input tensor shape! '
          'Expected: [1, width, height, 3]')
    required_image_size = (input_tensor_shape[1],
                           input_tensor_shape[2])
    with img.resize(required_image_size,
                    Image.NEAREST) as resized:
      input_tensor = np.asarray(resized).flatten()
      # Run model on accelerator
      return self.RunInference(input_tensor)[1]


class kNNEmbeddingEngine(EmbeddingEngine):
  """Extends embedding engine to also provide
     kNearest Neighbor detection. This class
     maintains an in-memory store of embeddings and
     provides functions to query with a new query
     embedding and find nearest neighbors.
  """

  def __init__(self, model_path, kNN=3):
    """Creates a kNNEmbeddingEngine with given model
    Args:
      model_path: path to TF-Lite Flatbuffer file.
      kNN: how many nearest neighbors to consider.
    """
    EmbeddingEngine.__init__(self, model_path)
    self.clear()
    self._kNN = kNN

  def clear(self):
    """Forgets all stored embeddings."""
    self._labels = []
    self._embedding_map = defaultdict(list)
    self._embeddings = None

  def addEmbedding(self, emb, label):
    """Add an embedding vector to the store."""
    # Normalize the vector
    normal = emb/np.sqrt((emb**2).sum())
    # Add to store, under "label"
    self._embedding_map[label].append(normal)

    # Expand labelled blocks of embeddings for when
    # we have less than kNN examples. Otherwise
    # blocks that have more examples unfairly win.
    emb_blocks = []
    self._labels = []
    for label, embeds in self._embedding_map.items():
      emb_block = np.stack(embeds)
      if emb_block.shape[0] < self._kNN:
          pads = [(0,self._kNN - emb_block.shape[0]),
                  (0,0)]
          emb_block = np.pad(emb_block, pads,
                             mode="reflect")
      emb_blocks.append(emb_block)
      self._labels.extend([label]*emb_block.shape[0])
```

```
        self._embeddings = np.concatenate(emb_blocks,
                                          axis=0)

    def kNNEmbedding(self, query_emb):
        """Returns the self._kNN nearest neighbors to a
           query embedding."""

        # If we have nothing stored, the answer is None
        if self._embeddings is None: return None

        # Normalize query embedding
        norm = np.sqrt((query_emb**2).sum())
        query_emb = query_emb/norm

        # We want a cosine distance from query to each
        # stored embedding. A matrix multiplication can
        # do this in one step, resulting in a vector of
        # distances.
        dists = np.matmul(self._embeddings, query_emb)

        # If we have less than self._kNN distances we
        # can only return that many.
        kNN = min(len(dists), self._kNN)

        # Get the N smallest distances.
        n_argmax = np.argpartition(dists, -kNN)[-kNN:]

        # Get the corresponding labels associated with
        # each distance.
        labels = [self._labels[i] for i in n_argmax]

        # Return the most common label over all
        # self._kNN nearest neighbors.
        most_common_label = Counter(labels)
        return most_common_label.most_common(1)[0][0]

    def exampleCount(self):
        """Returns the size of the embedding store."""
        return sum(len(v) for v in
                   self._embedding_map.values())
```



De USB Accelerator maakt deel uit van Corals lijn van AI-gerelateerde producten

Alle code die je nodig hebt om de machine te draaien vind je in deze directory. De code **embedding.py** op de vorige pagina's demonstreert de meest kritieke delen. Met de volgende opdracht test je de schakeling:

```
sudo python3 teachable.py --testui
```

De leds zullen knipperen om aan te duiden dat de schakeling werkt. Als je op de knoppen drukt, toont de Terminal om welke knop het gaat (tussen 0 en 4). **Druk op Ctrl+C** om het programma te onderbreken.

10 Herkennen maar!

Dan is het nu tijd om onze machine te draaien en te zien waartoe hij in staat is. Dat doe je met:

```
sh run.sh
```

De machine initialiseert zijn model en start dan. De led in de USB Accelerator zou nu aan moeten gaan en de Terminal toont een framerate (meestal rond de 30 fps). Hou een voorwerp (zoals een stuk fruit of een computermuis) boven de camera en druk op een van de knoppen bij een led (de led naast de knop gaat dan aan). Laat de knop los en haal het voorwerp weer weg.

Wanneer je het voorwerp nu weer voor de camera houdt, zal de corresponderende led aan gaan. Haal verschillende voorwerpen voor de camera en druk op verschillende knoppen om de machine te trainen om elk voorwerp te herkennen. Je kunt het best één knop aan alleen de achtergrond toewijzen (dus zonder een voorwerp voor de camera te houden).

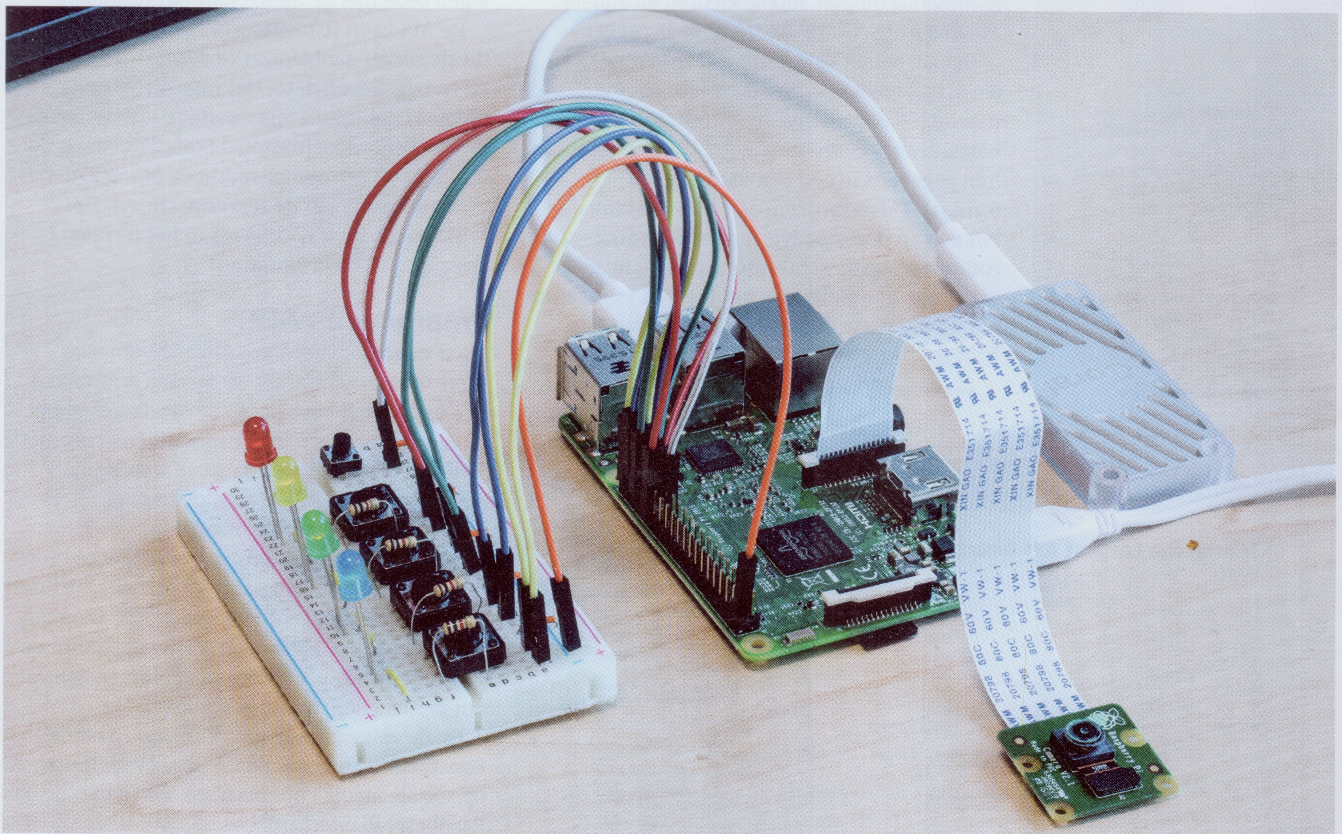
Als de machine wat onzeker lijkt, kun je altijd opnieuw op de knop drukken voor een hertraining. Druk meerdere keren op de knop en draai het voorwerp bij elke druk lichtjes.

Veel plezier met het herkennen van allerlei voorwerpen! Druk op de vijfde knop om alle voorwerpen uit het geheugen te verwijderen.

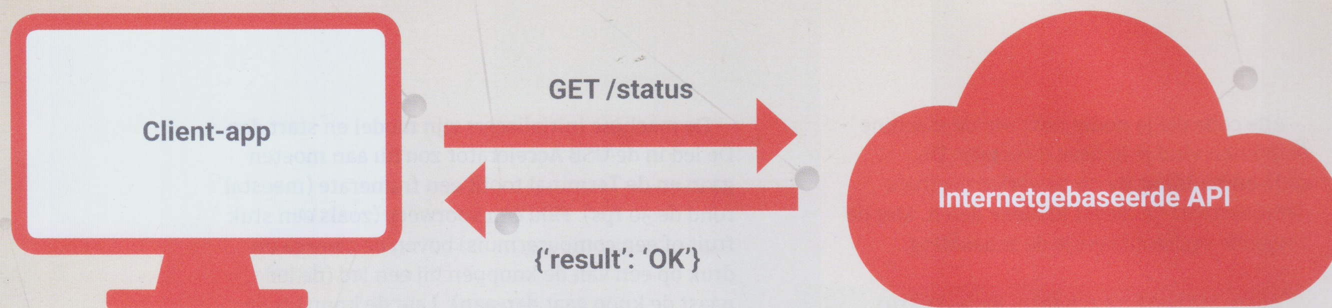
Tip 

Lasergesneden

Gebruik het bestand **rpi3plate.dxf** op onze GitHub-pagina als je een lasergesneden stabiel platform voor de componenten wilt maken.
magpi.cc/github



▼ De gemonteerde machine die we dingen kunnen aanleren. Wanneer je op een knop drukt, observeert het apparaat een object voor de camera. Dan herkent het hetzelfde object en laat het de bijbehorende led oplichten.



Sociale API's

Hoe communiceren computers over internet? **PJ Evans** introduceert je in de wereld van API's.

Wanneer wij mensen over internet willen communiceren, is een populaire manier om dat te doen een website maken die anderen kunnen bezoeken. Omdat we nu eenmaal mensen zijn, houden we van glimmende dingen, mooie kleuren en andere zaken die ons 'Oooh' en 'Ahh' doen zeggen.

Het zal je niet verrassen als we je zeggen dat speciale lettertypes, creatieve pagina-indelingen of je nieuwste minimalistische meesterwerk computers koud laten. Ze willen hun gegevens in een eenvoudig te lezen formaat. Dus terwijl wij op cookiemeldingen klikken en pop-upadvertenties wegklikken, zijn computers op de achtergrond op een heel wat eenvoudiger manier aan het kletsen — maar in plaats van via websites spreken ze via API's.

▲ API's gebruiken dezelfde protocols als het web (http) en geven data terug in een tekstformaat dat voor de client eenvoudig te ontleden is.

▼ Er bestaan duizenden API's om te verkennen. Op **programmableweb.com** vind je een lijst met de populairste.

Show apps | ProgrammableWeb | API DIRECTORY | API NEWS | WRITE FOR US | BECOME MEMBER | LOGIN
LEARN ABOUT APIS | WHAT IS AN API? | TUTORIALS | API CHARTS & RESEARCH | ADD APIs & MORE
MuleSoft | Turn existing services into APIs | Learn how

Search the Largest API Directory on the Web

Search Over 20,728 APIs | SEARCH APIS

Filter APIs: By Category | Include Deprecated APIs

API Name	Description	Category	Submitted
Google Maps	[This API is no longer available. Google Maps' services have been split into multiple APIs, including the Static Maps API,	Mapping	12.05.2005
Twitter	[This API is no longer available. It has been split into multiple APIs, including the Twitter Ads API, Twitter Search Tweets...	Social	12.08.2006

API UNIVERSITY — FEATURED | LATEST
FOR API PROVIDERS: What Are APIs and How Do They Work? · 8 Real World API Strategies and the Keys to Their Success · Microservices 101: Understanding and Leveraging Microservices · More for API Providers
FOR DEVELOPERS: How to Get Started With Google Actions · How to Build a Monitoring Application With the Google Cloud Vision API · How to Access Any RESTful API Using

Een API (*application programming interface*) is een verzameling afspraken op basis waarvan computerprogramma's met elkaar kunnen communiceren. Heel wat populaire webdiensten hebben API's, waardoor wij als gebruikers onze eigen code met de dienst kunnen koppelen.

De API geeft je toegang tot gegevens en acties die de server aanbiedt in de vorm van functies en teruggegeven data. Het meest voorkomende type API dat je zult tegenkomen, is gebaseerd op REST (REpresentational State Transfer). Dat is een heel eenvoudige methode om instructies en aanvragen naar de server te sturen. Het antwoord krijg je doorgaans in het formaat JSON (JavaScript Object Notation).

Waarom een API?

Maar waarvoor gebruik je API's? Noem elke grote dienst op internet en de kans is groot dat er een API voor bestaat. Als je een Twitter-app op je smartphone opent, communiceert die met Twitter via een API in plaats van webpagina's op te vragen. Dat werkt sneller en spaart heel wat bandbreedte uit. De meeste apps op je smartphone die gegevens van een externe bron benaderen, gebruiken daarvoor een API, inclusief je bank, Amazon, Facebook enzovoort. Gelukkig zijn veel van die API's open zodat wij ze kunnen gebruiken en er veel plezier aan kunnen beleven om ze met gelijk wat te koppelen.

Op de volgende pagina's kijken we naar de Fortnite API en hoe je daarmee je overwinningen op sociale media kunt plaatsen. Met een Raspberry Pi uiteraard!

Programmeer een lichtkrant voor FORTNITE met de Sense HAT

Hou je van Fortnite? Met onze lichtkrant toon je de laatste scores, nieuwsberichten en gebeurtenissen live op een Sense HAT.



PJ Evans

PJ is auteur, software-ingenieur en organisator van een Raspberry Jam. Momenteel verbergt hij zich voor iemand met een tomaat als hoofd.

mrpjevans.com

@mrpjevans

Je hebt nodig

- Sense HAT
- Gebruikersaccount bij Epic Games (**epicgames.com**)
- Internetverbinding
- Loot Llama-knuffel

Fortnite! Tenzij je het afgelopen jaar op Mars geleefd hebt of een heel productief leven geleid hebt, heb je zeker al over dit schietspelletje gehoord. In het speltype Battle Royale neem je het tegen 99 andere spelers op. Al wat je moet doen, is als laatste overblijven. Omdat Fortnite een constant evoluerend multiplayerspel is, zijn er talloze gegevens beschikbaar over spelerstatistieken, uitdagingen en spullen die te koop zijn.

Gelukkig bestaat er een eenvoudige API die al die informatie aanbiedt. We gebruiken een Sense HAT en Node-RED om een lichtkrant te maken voor de meest toegewijde Fortnite-spelers.

01 Prepare your Raspberry Pi

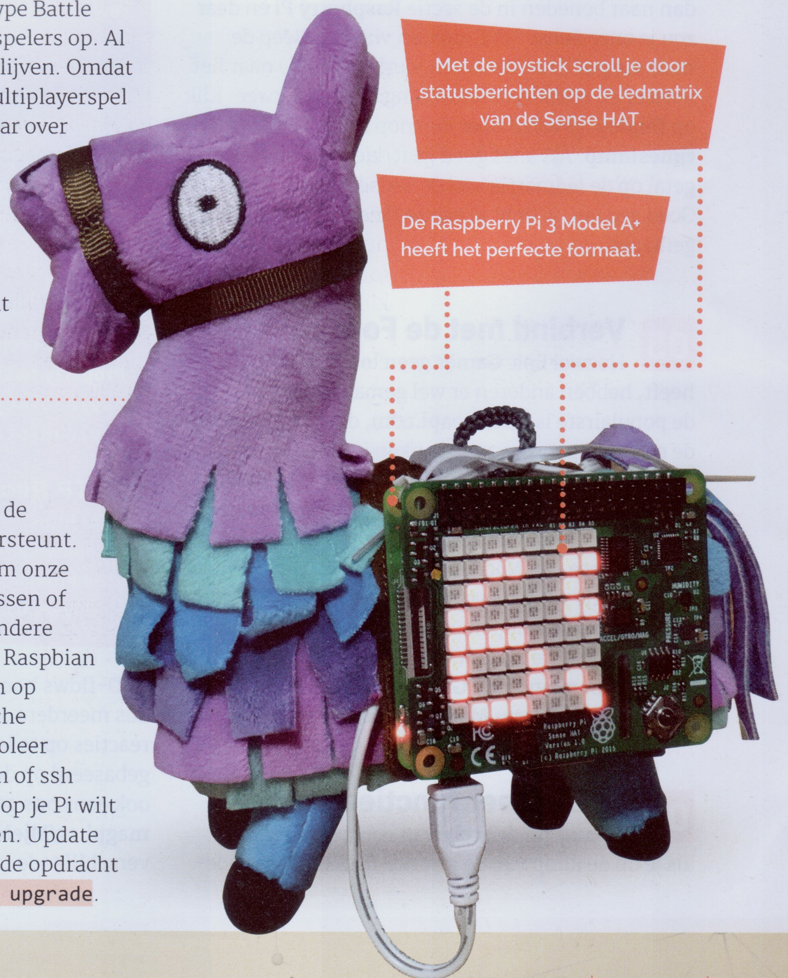
Dit project werkt met elk model van de Raspberry Pi dat de Sense HAT ondersteunt. We hebben een webbrowser nodig om onze code te creëren, dus je dient te beslissen of je alles op de Pi zelf doet of op een andere machine. In het tweede geval kun je Raspbian Stretch Lite als je besturingssysteem op de Pi installeren, omdat je de grafische interface dan niet nodig hebt. Controleer of de Pi met internet verbonden is en of ssh draait als je via een andere machine op je Pi wilt inloggen om opdrachten uit te voeren. Update daarna alle aanwezige software met de opdracht `sudo apt update && sudo apt -y upgrade`.

02 Node-RED

We gebruiken Node-RED om de code voor ons Fortnite-apparaat te genereren en uit te voeren. Deze geweldige ontwikkelomgeving kun je omschrijven als Scratch voor Internet-of-Things-apparaten en ze is bedrieglijk krachtig. Als je de volledige Raspbian geïnstalleerd hebt met alle aangeraden software, dan vind je Node-RED in het applicatiemenu onder **Programming**. Is Node-RED nog niet geïnstalleerd, volg dan de instructies op **magpi.cc/eHkkjz**. Nadat je het geïnstalleerd hebt, dien je er nog voor te zorgen dat Node-RED altijd opstart als je Pi opstart:

```
sudo systemctl enable nodered.service
```

Om dit te testen, herstart je de Raspberry Pi, open je een webbrowser en bezoek je **http://IP:1880** (met IP gelijk aan het ip-adres van de Pi). Op de Raspbian-desktop kun je ook **http://localhost:1880** bezoeken.



Met de joystick scroll je door statusberichten op de ledmatrix van de Sense HAT.

De Raspberry Pi 3 Model A+ heeft het perfecte formaat.

03 Installeer de node voor Sense HAT

Programma's heten in Node-RED flows en ze bestaan uit nodes die specifieke acties uitvoeren en de resultaten van die actie (het 'bericht') aan de volgende node doorgeven. Node-RED komt met een bibliotheek van nodes om invoer, uitvoer en stappen daartussen af te handelen. Voor de interactie met de Sense HAT dienen we een extra node te installeren en wat andere bibliotheken. Voer de volgende regels in de Terminal in:

```
cd ~/.node-red
sudo apt install -y python-pip sense-hat
sudo pip install pillow
npm install node-red-node-pi-sense-hat
node-red-restart
```

04 Je eerste flow

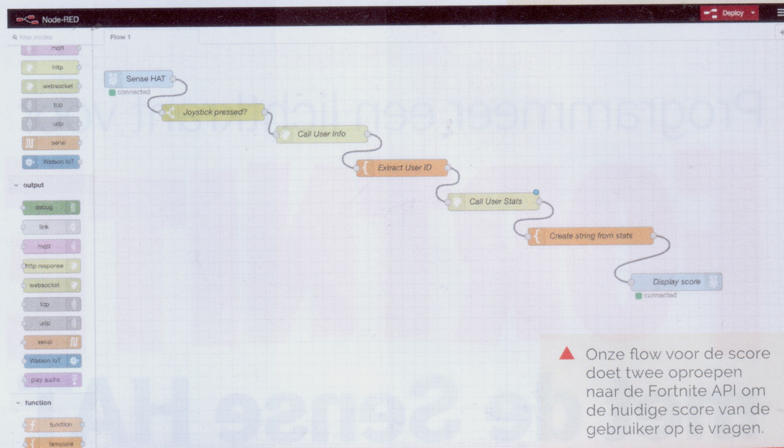
Flows dienen met een *trigger* te starten: een gebeurtenis die de flow start. Sleep van de lijst met nodes aan de linkerkant **inject** naar het rooster in het centrum (het verandert naar **timestamp**). Scroll dan naar beneden in de sectie **Raspberry Pi** en daar zou je onze Sense HAT moeten vinden. Sleep de node **output** (met een klein vierkant links) naar het rooster en maak een verbinding tussen de twee. Klik op **Deploy**, gevolgd door de knop links van de node **timestamp**. Als alles goed gaat, zie je nu een groot getal op de ledmatrix van de Sense HAT scrollen. Goed gedaan, je hebt je eerste flow in Node-RED gemaakt.

05 Verbind met de Fortnite API

Hoewel Epic Games geen 'officiële' API heeft, hebben anderen er wel gemaakt. Een van de populairste is **fortniteapi.com**, die ook een van de eenvoudigste om te gebruiken is. We doen twee http-aanroepen naar de API om de gebruikersdetails te krijgen, dan zoeken we de huidige score van de gebruiker op en tonen die op de Sense HAT. Open de webinterface van Node-RED zoals tevoren, verwijder de nodes die we in de vorige stap toegevoegd hadden en doorloop de stappen in het kader hiernaast om je nieuwe flow te maken. Een alternatief is dat je de flow van **magpi.cc/PGvDBa** downloadt en dan inlaadt via het menu **Menu / Import / Clipboard.**

06 Voeg meer functies toe

Je kunt nu de score van een gebruiker zien als je op de knop van de joystick drukt. Maar Node-RED-flows hoeven niet lineair te zijn. Je kunt dus meerdere acties verwerken om verschillende reacties op te leveren en je code te vertakken gebaseerd op de inkomende gegevens. Bekijk ook eens ons geavanceerdere Python-script op **magpi.cc/MjeNBe**: zo zie je dat je dezelfde API in verschillende talen kunt gebruiken.



▲ Onze flow voor de score doet twee oproepen naar de Fortnite API om de huidige score van de gebruiker op te vragen.

score.json

➤ Taal: **Node-RED**

DOWNLOAD DE VOLLEDIGE CODE:



magpi.cc/MjeNBe

Om je Fortnite-flow te creëren, doorloop je elke stap hieronder, sleep je de vereiste node naar het rooster, verbind je het met de vorige node en ga je zo van links naar rechts. Dubbelklik op de node om de eigenschappen in te stellen en de verschillende formuliervelden zoals gegeven in te vullen. Laat alle andere velden op hun standaardwaarde.

Sense HAT Input
Output: Joystick events only

Switch
Name: Joystick pressed?
Property drop-down: Expression
Property field:
`payload[key='ENTER'] and payload[state=0]`

HTTP Request
URL: https://fortnite-public-api.theapinetwork.com/prod09/users/id?username=USERNAME
(N.B. Vervang USERNAME door je gebruikersnaam bij Epic Games.)
Return: a parsed JSON object
Name: Call User Info

Template
Name: Extract User ID
Set property: url (Laat de drop-down op 'msg' staan.)
Template: `https://fortnite-public-api.theapinetwork.com/prod09/users/public/br_stats?user_id={{payload.uid}}&platform=pc`

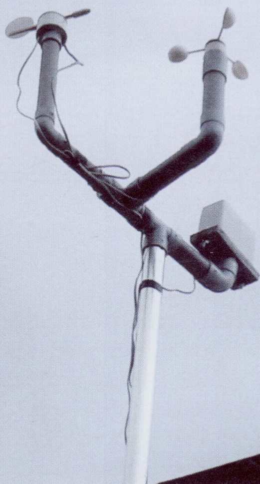
HTTP Request
Return: a parsed JSON object
Name: Call User Stats

Template
Name: Create string from stats
Template:
`Solo Score: {{payload.stats.score_solo}}`

Sense HAT Output
Name: Display score

Publieke API-projecten om te bekijken

Publiek beschikbare API's bieden toegang tot een overweldigende hoeveelheid informatie. Hier volgen enkele van onze favoriete Pi-gebaseerde projecten.



WEERBERICHT

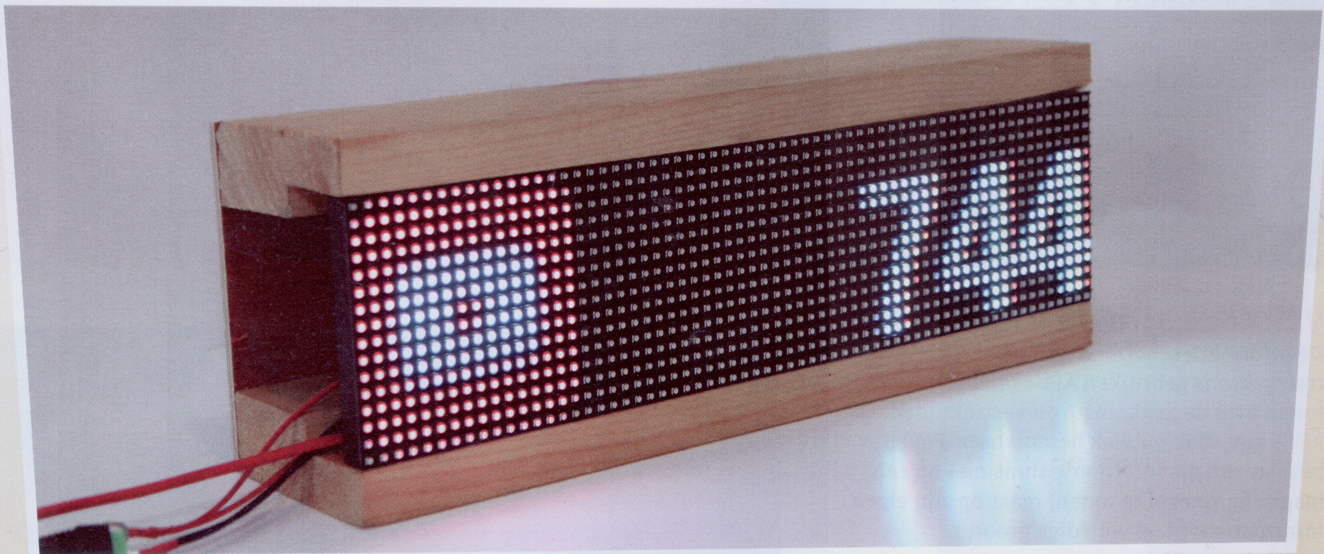
Deze uitleg is een geweldige manier om kennis te maken met het programmeren met een API. Met deze API die toegang geeft tot weerstations in scholen kun je live weergegevens ophalen in slechts enkele regels Python-code. Op welke leuke manier zou je de resultaten kunnen tonen?

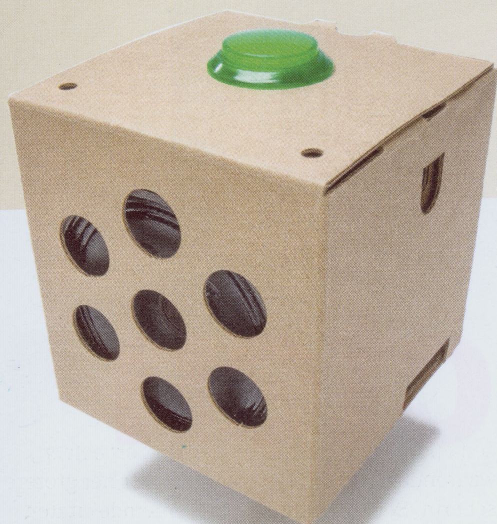
> magpi.cc/FvdhbK

TEL JE VOLGERS OP SOCIALE MEDIA

Dit project is wat geavanceerder, maar het toont je dat API's verbinden niet alleen mooi maar ook praktisch kan zijn. Je bouwt een grote ledmatrix met twee schermen om het aantal volgers op sociale media met meerdere API's op het scherm voorbij te laten schuiven. Ideaal voor een kantoor of gewoon om thuis op te scheppen.

> magpi.cc/WhVLNA





AIY VOICE KIT

Een ander voordeel van API's is de mogelijkheid om moeilijk werk uit te besteden aan een server 'ergens in de cloud'. De AIY Voice Kit van Google uploadt audio van de microfoon naar de servers van Google via een API, om alle ingewikkelde analyses daarop uit te voeren die een Raspberry Pi zouden overweldigen.

> aiyprojects.withgoogle.com



Image Credit: Lucas Berbesson

NOTIFICATIEDOOSJE

Dit mooie doosje licht op wanneer je een notificatie krijgt op YouTube, Facebook of Instagram. De lichtkleur verandert op basis van welke dienst de notificatie stuurt. En als een bonus krijg je ook een applausje! Weer een geweldige manier om meerdere API's in één apparaat samen te brengen.

> magpi.cc/uOnUvG

Woordenlijst

Zoals de meeste technologie is de wereld van API's doorspekt met jargon. PJ Evans komt je ter hulp.

REST

REpresentational State Transfer. Een de facto standaard voor hoe API's werken. Opdrachten stuur je over http met de werkwoorden GET, PUT, POST, PATCH en DELETE waarmee je het type actie aanduidt.

SOAP

Simple Object Access Protocol. Een ouder systeem om opdrachten uit te sturen, vooral Microsoft is er voorvechter van.

JSON

JavaScript Object Notation. De meest voorkomende manier om data in API-transacties voor te stellen. Beschouw het als een geavanceerdere versie van csv-bestanden die ook geneste gegevens ondersteunt. Op **json.org** vind je meer details.

YAML

Yet Another Markup Language of YAML Ain't a Markup Language. YAML is een concurrent van JSON die voor mensen zelfs gemakkelijker is om samen te stellen en te lezen. Net zoals Python vertrouwt het op inspringingen in de code.

XML

eXtensible Markup Language. Een platformonafhankelijke manier om gegevens voor te stellen. Hoewel JSON en YAML de fakkel overgenomen lijken te hebben, is XML nog altijd heel populair, en het is een broertje van HTML.

EINDPUNT

Een eindpunt (*endpoint*) is de basis-url die je gebruikt voor toegang tot een API. Verschillende componenten in die API zijn paden die je vanaf het eindpunt bereikt.

API- EN GEBRUIKERSSLEUTEL

Om beveiliging aan te bieden, vereist een API vaak dat je een API-sleutel en een gebruikerssleutel met je aanvragen meelevert. De gebruiker is aan je account gekoppeld en de API-sleutel aan je toepassing.

OAUTH

OAuth 2 is een geavanceerdere beveiligingsmethode voor API's. Dat vereist dat de server expliciete toestemming van de gebruiker vraagt voordat ze toegang tot de API-client geeft. Na de toestemming, wat doorgaans manueel gebeurt, krijgt de client een token dat je voor de toegang erna gebruikt.

OAUTH TOKEN

Als een API OAuth 2 voor zijn beveiliging implementeert, moet de API-client zijn unieke geheime sleutel gebruiken om een token te krijgen: een tekenreeks die als een wachtwoord dient en uniek is voor die sessie.

VERKEERSINFORMATIEBORD

De informatieschermen in bushokjes en op treinperrons gebruiken API's om berichten te tonen over de aankomst- en vertrektijden. Die API's zijn vaak publiek beschikbaar. Paul Shved besliste om die informatie thuis niet op een magische spiegel te tonen, maar op zijn eigen informatiebord op volledige grootte.

> magpi.cc/ovSjxk



Image Credit: Paul Shved

Inky wHAT

▶ Pimoroni ▶ **www.elektor.nl/black-white-inky-what** ▶ € 52,95

Dit grotere e-ink-scherm biedt meer ruimte voor afbeeldingen en tekst. Door **Phil King.**

What is een wHAT? Heel eenvoudig: een wide HAT. Het 4,2 inch-scherm van de Inky wHAT is 4,7 keer zo groot als de Inky pHAT en heeft 5,4 keer zoveel pixels. Dat maakt het stukken beter om gedetailleerde afbeeldingen en langere tekst te tonen. Een kleine vrouwelijke header op de achterkant lust tien gpio-pennen door, inclusief I²C en SPI — handig om sensoren en andere componenten aan te sluiten.

Gemonteerd scherm

We hebben de zwart-rood-witte versie getest (er is ook een zwart-witte versie). Omdat de wHAT volledig geassembleerd is, dien je hem alleen nog maar op je Pi te bevestigen met de meegeleverde afstandhouders en een gpio-headeruitbreider om het bordje te verhogen op de grote Pi-modellen. Maar wees voorzichtig en zet niet te veel kracht op het breekbare glas van het scherm. De Inky Python-bibliotheek installeer je met één opdracht, samen met de codevoorbeelden, waaronder zelfs enkele specifiek voor het grotere scherm. Een programma met beroemde citaten toont je hoe je tekst op de volgende regel laat doorlopen om op het scherm te passen. Een ander script toont je hoe je afbeeldingen van grootte verandert en het kleurenpalet aanpast. Dat werkt het beste met eenvoudige afbeeldingen, foto's zijn nog wel herkenbaar maar een beetje korrelig. De beste resultaten verkrijg je als je je eigen afbeeldingen in GIMP maakt en een geïndexeerd kleurenpalet gebruikt.

> Het scherm heeft ongeveer 25 seconden nodig om het hele beeld te vernieuwen, waardoor het niet voor alle projecten geschikt is

Het scherm heeft ongeveer 25 seconden nodig om het hele beeld te vernieuwen. Dat gebeurt met een boel geflikker, waarna de afbeeldingen geleidelijk verschijnen. Het is daardoor echt alleen maar geschikt voor projecten die maar zo nu en dan een vernieuwing van het scherm nodig hebben. Het voordeel is dan weer dat het — omdat het om een e-ink-scherm gaat — alleen stroom verbruikt tijdens het vernieuwen van het scherm.

SPECS

SCHERM: 4,2 inch EPD, 400×300 pixels

BRUIKBARE SCHERMOPPERVLAKTE: 84,8×63,6 mm

WERKT OP: Elke 40-pins Raspberry Pi

Verdict

Door de lange schermvernieuwingstijd is de Inky wHAT niet voor elk project bruikbaar. Maar het grotere scherm is ideaal voor een weersvoorspelling, kalender, nieuwsberichten of todo-lijstje.

8/10



▼ De Inky wHAT doet de pHAT er als een dwerg uitzien: hij biedt heel wat meer scherm.

▲ De extra header op de achterkant lust tien van de gpio-pennen van de Pi door.

TinyPi Pro

▶ Pi0cket ▶ pi0cket.com ▶ € 80

De kleinste retrospelconsole die je ooit zag, en je kunt ze zelf maken. **Rob Zwetsloot** haalt zijn Pi Zero boven.

SPECIFICATIES

SCHERM:
1240 × 240 pixels, 1,3"

BATTERIJ:
Herlaadbare 400 mAh

AFMETINGEN:
69 × 34 × 20 mm

POORTEN:
HDMI uit, USB in

OPSLAG:
Verwijderbare microSD-kaart

Meer dan een jaar geleden was de TinyPi nog een klein interessant hobbyprojectje. De maker Pete Barker heeft nu een kleine kit samengesteld zodat je zelf ook een extreem kleine spelconsole in elkaar kunt zetten.

En we bedoelen echt extreem klein — het is ontworpen om even klein als een Pi Zero te zijn, zij het bijna 20 mm diep. Het is dan ook een vrij opmerkelijke kit: er passen een D-pad, zes knoppen van voren en twee schouderknoppen in, terwijl het ook nog een scherm en luidsprekers in de behuizing heeft. Dat is indrukwekkend!

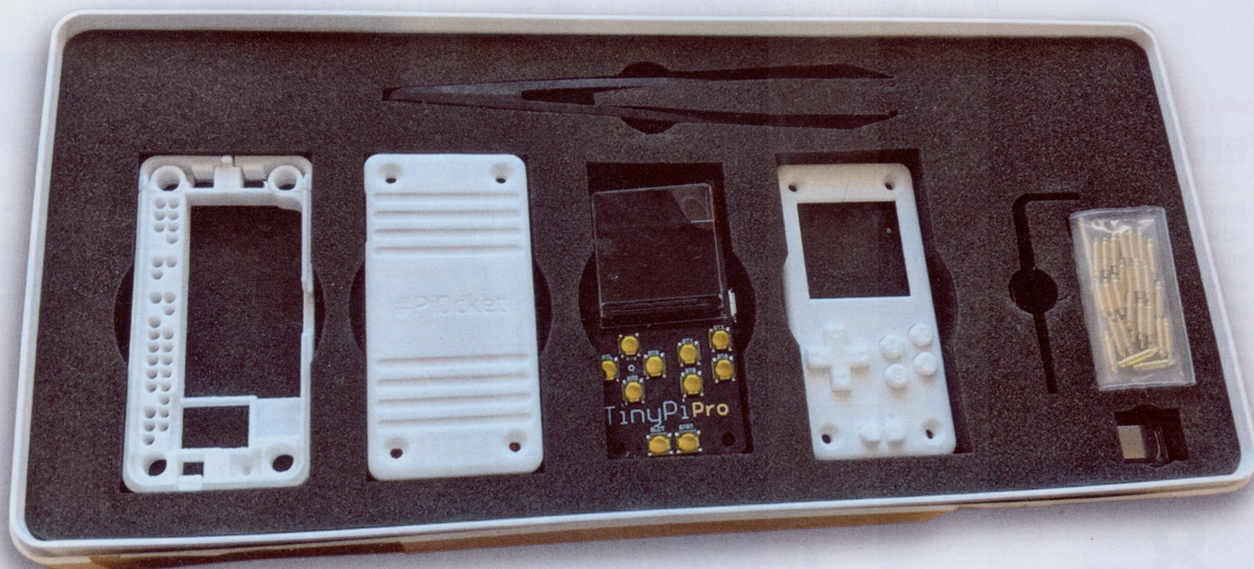
Het komt in de vorm van een kit. Daar dien je je eigen Pi Zero aan toe te voegen (de gpio-pennen zijn niet nodig), evenals een microSD-kaart om een besturingssysteem op te installeren.

Voor een projectkit van deze kleine afmetingen dien je meestal zelf nog wat te solderen om alles te laten passen. Met de TinyPi niet — de enige bevestigingselementen die je nodig hebt, zijn vier schroeven die je met vier afstandhouders verbindt, en een inbussleutel om ze vast te draaien is in de kit meegeleverd.

Klein gebouwd

Met iets van deze kleine afmetingen beeld je je al in dat het best wel lastig is om in elkaar te steken. En het pincet dat in de kit meegeleverd is, doet die angst alleen maar toenemen. Maar tot onze verbazing was het enige echte lastige onderdeel het aansluiten van de (nogal kleine) batterij. Het pincet werkte daar overigens perfect voor. Al de rest viel, gleed of klikte gewoon op zijn plaats.

De instructies doen het uitschijnen dat het eenvoudig is om de kit in elkaar te steken, en voor het grootste deel is dat ook zo. Jammer genoeg



▲ De kit komt mooi in een doos, met alles zichtbaar.

worden er enkele zaken in de uitleg verwaarloosd. We hopen dat deze in de online uitleg wat meer uitgewerkt worden

Klein vermogen

Dankzij de combinatie van RetroPie en de kracht van een Pi Zero, heeft de TinyPi Pro de middelen om heel wat geëmuleerde en homebrew games te spelen. RetroPie configureren is eenvoudig, maar je dient wel te weten hoe je de controllerconfiguratie kunt overslaan nadat alle beschikbare knoppen gebruikt zijn. Tip: houd gelijk welke knop ingedrukt.

Hoewel er meer dan genoeg knoppen zijn voor de meeste games die je op de TinyPi Pro kunt spelen, is het spelen zelf met lompe volwassen handen best moeilijk. Vingers en duimen zitten elkaar in de weg, en al snel is je hand verkrampt door de houding die nodig is om de schouderknoppen te kunnen bereiken. We ontdekten dat spelletjes die niet zoveel knoppen nodig hebben heel wat beter werkten, maar kinderen met kleinere handen zullen zeker van alle games kunnen genieten.

Het kleine scherm is in orde, en je zult niet snel een spel vinden waarmee je problemen hebt om alles op het scherm te onderscheiden.

Het is een mooie kit die leuk is om te bouwen. Het uiteindelijke resultaat is indrukwekkend, maar het is niet de beste console om spelletjes op te spelen. Maar als je ergens onderweg bent en zin hebt in een retrospelletje, is het handig dat de TinyPi Pro in je broekzak past.



▲ Vergeleken met enkele tijdgenoten is het een ongelooflijk klein apparaat.



◀ Hoewel de afmetingen heel handig zijn, is het daardoor wel een beetje ongemakkelijk om te spelen.

> "Spelletjes die niet zoveel knoppen nodig hebben werken heel wat beter"

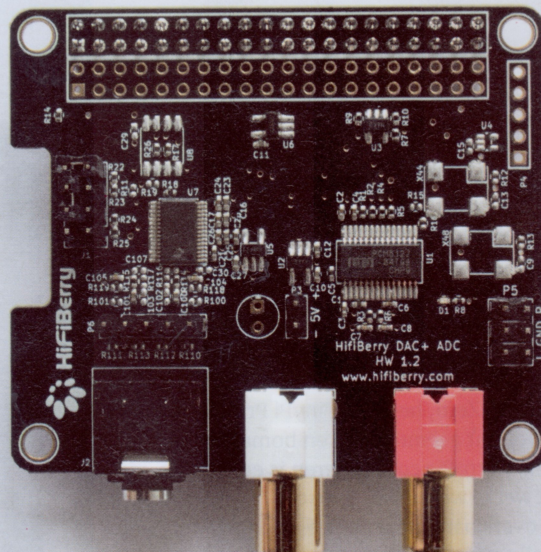


▲ De kleine TinyPi Pro is niet zo gemakkelijk om vast te houden. De banaan is voor de schaal.

Verdict

Een echt keurige kit die leuk is om te bouwen en een geweldig eindresultaat oplevert. Het is een beetje te klein voor grote volwassen vingers, maar voor jongere spelers is het goed geschikt.

8/10



De DAC+ ADC is een HAT die je eenvoudig op de gpio-header van de Pi schuift, zonder dat je iets extra hoeft aan te sluiten.

HiFiBerry DAC+ ADC

HiFiBerry ▸ magpi.cc/NshChD ▸ € 53

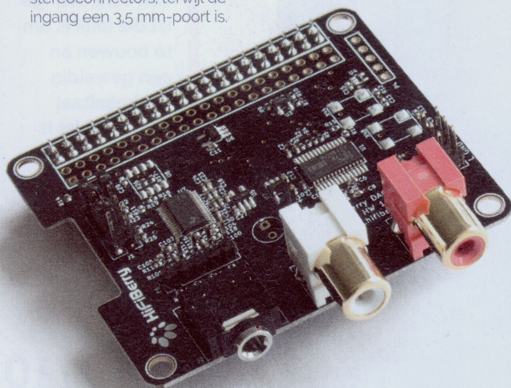
Er bestaan talloze HAT's met audio-uitgangen voor de Raspberry Pi, legt **K.G. Orphanides** uit, maar deze heeft ook een ingebouwde analoog-digitaalomzetter om audio op te nemen.

In tegenstelling tot de meeste Raspberry Pi audio-HAT's heeft de HiFiBerry DAC+ ADC een analoge audio-ingang. Daardoor kun je zowel geluid opnemen als afspelen. Perfect voor compacte audioproductieprojecten dus.

De fysieke aansluiting is eenvoudig, maar je hebt een Linux 4.18.12-kernel nodig om de analoog-digitaalomzetter (adc) te gebruiken. Voor de volledige instructies om je kernel te upgraden en de opstartconfiguratie aan te passen verwijzen we je naar de datasheet van de DAC+ ADC op de website van HiFiBerry.

De Burr-Brown PCM5122 DAC-chip is een populaire keuze voor redelijk geprijsde computeraudiohardware, en met reden. Het is een plezier om ernaar te luisteren, zeker door hogekwaliteitsluidsprekers via de stero-RCA-uitgang. Het bordje heeft wel geen ingebouwde hoofdtelefoonversterker.

De adc — een Burr-Brown PCM1861 — heeft een 3,5 mm stereo-ingang. Deze is standaard geconfigureerd om lijnniveau-audio aan te nemen, zoals wat er uit je smartphone of de lijnuitgang-connectors op de meeste audioapparatuur komt. Je kunt dit gebruiken om analoge media zoals cassettebandjes te digitaliseren, om van je Pi een draagbaar doosje met instrumenteffecten te maken, of om van een mengpaneel op te nemen.



De audio-uitgang verloopt via een paar RCA-stereoconnectors, terwijl de ingang een 3,5 mm-poort is.

> "Gebruik de adc om je cassettebandjes te digitaliseren, om van je Pi een draagbaar doosje met instrumenteffecten te maken, of om van een mengpaneel op te nemen"

Lofzang

Je kunt ook dynamische microfoons aansluiten — we hebben dit met een Shure SM58 getest — als je een jumperschakelaar aanpast om 32 dB versterking in te schakelen. Het werkt perfect voor opnames van zang, karaokefeestjes of zelfs om audio-passthrough in software in te schakelen om een geïmproviseerde omroepinstallatie te maken. Merk op dat de DAC+ ADC geen fantoomvoeding kan leveren die echte condensatormicrofoons van studiokwaliteit of compacte elektretmicrofoons nodig hebben. Extra headers op het bordje laten je toe om externe versterkers en gebalanceerde ingangen aan te sluiten.

SPECS

HARDWARE: DAC: Burr-Brown PCM5122 (192 kHz/24-bits), ADC: Burr-Brown PCM1861 (192 kHz/24-bits)

SNR: DAC: typisch 112 dB, ADC: typisch 110 dB

POORTEN: 3v5 mm audio in, stereo RCA uit, gebalanceerde ingangsheader, uitgangsheader

Verdict

De DAC+ ADC handelt verliesloze muziek, games, midi-softsynths en geluidsproductie briljant af. Als je een compact bordje wilt om volledige audio-functionaliteit aan je Pi toe te voegen, zoek dan niet verder.

9/10



Coolest Projects Belgium

Jonge makers stellen hun technologische projecten voor

Misschien woont de volgende Grace Hopper of Elon Musk wel in België. Dat is het idee achter Coolest Projects Belgium, een technologiebeurs voor jonge uitvinders van 7 tot 18 jaar die jaarlijks door CoderDojo Belgium met de steun van Telenet georganiseerd wordt. Het concept bestaat al even in Ierland, waar je sinds enkele jaren naar een gigantische uitvindersbeurs voor kinderen en jongeren kunt. In 2016 werd beslist om een Belgische tegenhanger te organiseren. Een schot in de roos, want het evenement wint elk jaar aan populariteit.

Op 27 april was Coolest Projects al aan zijn vierde editie toe. Meer dan honderd jonge makers waren voor de gelegenheid naar Technopolis Mechelen afgezakt om hun nieuwste creaties voor te stellen aan het grote publiek. Van grappige computergames tot dolgedraaide robots, niets was aan de creativiteit van de deelnemers ontsnapt. Hierbij een greep uit hun projecten.

De automatische parkeergarage van Natan Henau (9 jaar)

Natan leerde bij CoderDojo met Arduino werken en was meteen verkocht. Een van zijn favoriete vrijetijdsbestedingen is om allerhande installaties te maken met zijn Arduino.

Voor Coolest Projects wilde hij zijn legostad de 21ste eeuw binnenloodsen. Daarom bouwde hij een hoogtechnologische autogarage met automatische lift. Echt indrukwekkend is de slagboom die alleen opengaat wanneer Natan zijn vinger voor de scanner houdt.

Daarvoor lijmde hij een vingerafdrukscanner op een legoblokje en programmeerde hij vervolgens zijn Arduino zodat die alleen reageert wanneer zijn vingerafdruk herkend wordt. Via een servomotor laat de Arduino vervolgens een slagboom 4 seconden openen, zodat de legowagens de parkeergarage kunnen binnenrijden.


Natan en zijn 21ste-eeuwse parkeergarage

De toverspiegel van Wannes en Robbe Ghyssaert (12 en 10 jaar)

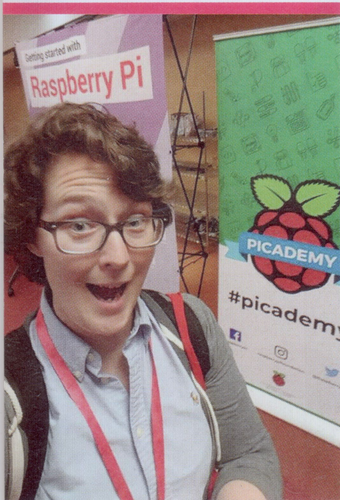
Broers Wannes en Robbe hadden een praktisch probleem voor ogen toen ze over hun project aan het brainstormen waren. Hun oma, die ze ten huize Ghyssaert 'moeke' noemen, woont wat ver weg. In tijden van webcams en FaceTime hoeft dat geen probleem te vormen, maar moeke houdt niet zo van technologie. Laptops en smartphones laat ze steevast links liggen.

Daarom kwamen Wannes en Robbe met een ingenieus idee. Ze werkten een hedendaagse versie van de spiegel van sneeuwwitje uit die hen in staat stelde om met hun moeke te praten, zonder dat ze daarvoor een technologisch apparaat in huis moet halen.

De jonge uitvinders bouwden daarvoor een kader rond een beeldscherm en monteerden er een spiegelglas voor. Daardoor krijg je een spiegeleffect, maar komt het licht van het beeldscherm er wel nog steeds door. Samen met een camera en een microfoon hebben ze het geheel verbonden met een Raspberry Pi. Vervolgens programmeerden ze in HTML/CSS en JavaScript de verschillende opties van de spiegel.

Overdag toont de Pi een startscherm waar het weer, nieuwsberichtjes en agenda's op gedeeld kunnen worden. Wanneer de jongens 's avonds na school thuiskomen, gaat de webcam aan en verschijnen ze op de spiegel. Zo houden Wannes en Robbe dagelijks contact met hun moeke via haar 'toverspiegel'.


Moeke voor haar toverspiegel



Lisa Rode

Robots in de klas maken de lessen niet alleen leuk, maar leren ook veel meer aan dan computerwetenschappen

> Category **Onderwijs** | > Job **Lerares** | > Twitter **@roderunners**

De Raspberry Pi Foundation heeft geweldige zaken gedaan om het informaticacurriculum in Britse scholen te hervormen. Onderwijzers in andere landen over de hele wereld hebben dat opgemerkt. Een van die onderwijzers is Lisa Rode, die in Springfield (Virginia, in de Verenigde Staten) lesgeeft in groep 8.

"In 2014 startte ik met een naschoolse roboticaclub met een Raspberry Pi. Ik leerde kinderen er met de GoPiGo-robotkit van Dexter Industries programmeren, ingenieursvaardigheden toepassen en problemen oplossen", vertelt Lisa ons. "Toen ik daardoor de kracht zag van het aanleren van vaardigheden in *physical computing* en computationeel denken, begon ik meer technologie en in het bijzonder robotica in mijn standaardlessen op te nemen. In de laatste vier jaar heb ik mijn klaslokaal getransformeerd tot een makerspace waar alle leerlingen de kans krijgen om hun begrip van technologie op allerlei manieren aan te tonen. Alle leerlingen maken in mijn lessen kennis met robotica, programmeren, 3d-modelleren, 3d-printing en nog meer."

Lisa geeft wiskunde, maatschappijleer, wetenschappen, lezen en schrijven en ze heeft in de laatste jaren in een heel diverse en inclusieve klas lesgegeven. Geïnspireerd door Picademy heeft ze een cursus gecreëerd om andere leerkrachten op haar school bij te leren over programmeren en physical





> De kinderen programmeren GoPiGo-robots in de taal Bloxter.

▼ Lisa heeft meerdere Raspberry Pi-werkstations opgezet voor haar lessen.


Lisa geeft Amerikaanse geschiedenis — in dit geval de westwaartse uitbreiding — met behulp van robots.


In Lisa's klaslokaal leren de leerlingen met heel wat leuke robots werken.


De Raspberry Pi's worden ook ingezet voor wetenschaps-experimenten, zoals dit experiment over broeikasgassen.

computing. Ze werkt ook samen met Dexter Industries aan hun curriculum.

Hoe ben je betrokken bij Dexters curriculum?

Ik schrijf een curriculum voor de robotkit van Dexter Industries. Een groot deel van wat ik doe, is lessen maken die in de standaardlessen te integreren zijn. In lessen over de plotstructuur van een verhaal leer ik de kinderen bijvoorbeeld hoe ze van de GoPiGo-robot een personage maken en het programmeren zodat het plot alle belangrijke elementen bevat: de expositie, oplopende actie, climax en teruglopende actie.
Een andere les focust op de Mars Rover-eenheid: leerlingen leren daar over ruimteverkenning, ruimtetechnologie en planeten in ons zonnestelsel. Daarna ontwerpen ze hun eigen Mars Rover met de GoPiGo als startpunt.
Ik werk ook samen met Nicole Parrot, de cto van Dexter Industries, om nieuwe blokken te bespreken voor Bloxter, een van de programmeertalen voor de GoPiGo-robots. Nadat ik die blokken zelf en in mijn klaslokaal uitgetest heb, komen ze beschikbaar voor iedereen. Een voorbeeld is het blok 'curve' dat we creëerden nadat mijn leerlingen met GoPiGo-robots modellen van de beweging van de Aarde rond de zon gecreëerd hadden.

> " Een groot deel van wat ik doe, is lessen maken die in de standaard-lessen te integreren zijn "

Welke achtergrond in computers heb je?

Ik heb geen enkele formele opleiding of achtergrond in computers. Ik kwam gewoon enkele jaren geleden wat informatie tegen over de Raspberry en ik besloot om er voor mezelf een te kopen. Nadat ik er wat mee had geëxperimenteerd, bedacht ik me dat het geweldig zou zijn om wat Pi's in mijn klaslokaal te hebben. Dat leidde uiteindelijk tot nog meer Raspberry Pi-werkstations in het lokaal en ook wat GoPiGo-robotkits.
Vorige zomer had ik het geluk om aan Picademy in Jersey City te kunnen deelnemen. Ik leerde er zoveel en ik bracht die informatie mee naar mijn school.

Uitgelichte projecten

PLANETAIR ROBOTWAGENTJE

"Een van mijn favoriete projecten is toen mijn leerlingen hun GoPiGo-robots transformeerden tot planetaire robotwagentjes. Dit jaar voegden we iets nieuws toe. Omdat NASA in juni 2018 door een stofstorm het contact met de Mars Opportunity verloor, voegde ik aan de missie van de leerlingen de taak toe om de Opportunity te helpen redden. De leerlingen bouwden en programmeerden hun robotwagentjes om de Opportunity te vinden en te redden."



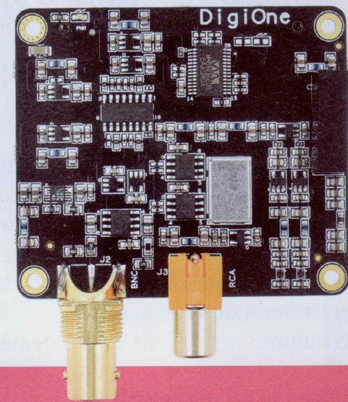
KARTONNEN ARCADESPEL

"Enkele jaren geleden creëerden mijn leerlingen een kartonnen arcadespel in de stijl van Pac-Man. Het hart bestond uit een Raspberry Pi en ze programmeerden het in Scratch. De aansturing creëerden ze met een Makey Makey."

REVIEW

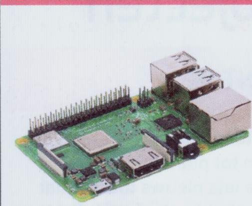
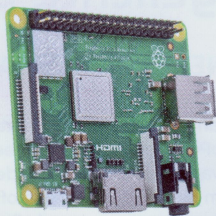
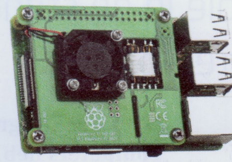
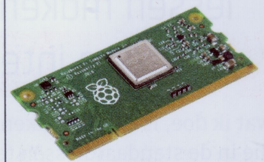
Allo DigiOne – Digitale audio-uitvoer (S/PDIF) voor de Pi

Voor de kostprijs van drie Raspberry Pi's krijg je van Allo de DigiOne: een HAT (Hardware Attached on Top) dat het audiosignaal dat op de Raspberry Pi afspeelt als een digitaal S/PDIF-signaal beschikbaar stelt. Waarom zou je dat willen? Nou, de DigiOne functioneert samen met de Raspberry Pi als een S/PDIF-transporter: een apparaat dat audiogegevens van een opslagapparaat (usb-stick, harde schijf, nas, internetstream enzovoort) digitaal naar de dac overbrengt.



Lees de volledige review op www.elektormagazine.nl/review-allo-digione

RPi Bestsellers

1. Raspberry Pi 3B+	2. Raspberry Pi 3A+	3. JoyPi	4. PoE HAT voor Raspberry Pi 3B+	5. RPi Compute Module 3+ (32 GB)
				
www.elektor.nl/rpi-3b-plus	www.elektor.nl/rpi-3a-plus	www.elektor.nl/joypi	www.elektor.nl/poe-hat-rpi-3-plus	www.elektor.nl/rpi-cm3-32gb

Boeken

Motor Control – Projects with Arduino & Raspberry Pi

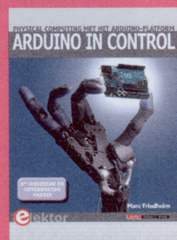
Dit Engelstalige boek gaat over dc-motors en hun gebruik in projecten met een Arduino of Raspberry Pi Zero W. Het staat vol met geteste en werkende projecten, van standaard dc-motors en stappenmotors tot servomotors en mobiele robots.



www.elektor.nl/motor-control

Arduino in Control

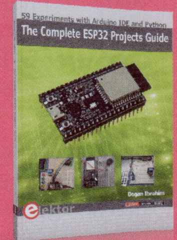
Dit leerboek over het Arduino-platform is uniek en brengt je stap voor stap van beginner tot gevorderde Arduino-gebruiker. Er wordt duidelijk in uitgelegd waarom en waardoor toepassingen werken, ondersteund door gedetailleerde afbeeldingen in kleur.



www.elektor.nl/arduino-in-control

The Complete ESP32 Projects Guide

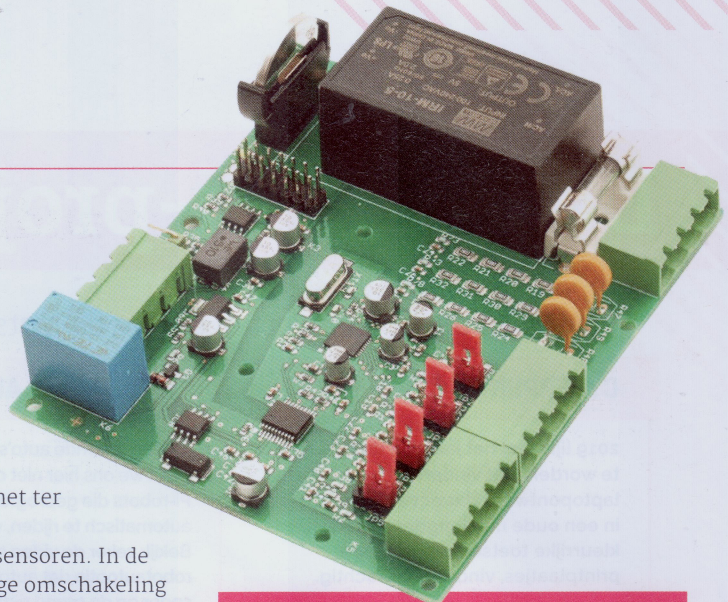
Het hoofddoel van dit Engelstalige boek is om de programmeertalen Arduino IDE en MicroPython uit te leggen voor ESP32-gebaseerde projecten. Het boek doet dat met het populaire ESP32 DevKitC-ontwikkelbordje met talloze projecten van eenvoudig tot gemiddeld niveau.



www.elektor.nl/esp32-projects-guide

In de kijker

SmartPi 2 inclusief 3 stroomprobes



De uitbreidingsmodule SmartPi 2 breidt de Raspberry Pi uit met interfaces om spanning en stroom op een contactloze manier uit te lezen. Het bordje laat toe om elektriciteitsproductie en -consumptie te controleren en maakt van de Raspberry Pi een volwaardige slimme meter. Alle meetgegevens kun je opslaan of via het lokale netwerk of internet ter beschikking stellen.

De stroom wordt contactloos gemeten via inductieve stroomsensoren. In de standaardversie zijn stromen tot 100 A te meten. Een eenvoudige omschakeling van een jumper laat een kabelomvormer met een tweede stroom van 1 A toe. Dat laat toe om stromen tot 700 A en meer te detecteren.

De SmartPi 2 heeft vier ingangen voor stroommetingen (L1, L2, L3 en N) en drie ingangen voor spanningsmetingen. Dat laat metingen op alle drie de fases toe en zelfs de neutrale geleider. De SmartPi 2 en de Raspberry Pi kunnen door de ingangsspanning gevoed worden. Een externe voeding is niet nodig.

Als de spanningsmeting verbonden is, kan de richting van de energiestroom bepaald worden en de SmartPi is dan ook een volledig uitgeruste bidirectionele slimme meter.

Andere eenheden:

Stroom · Spanning · Vermogen · Actief vermogen · Reactief vermogen · Schijnbaar vermogen · Energieverbruik · Energieopwekking · Vermogenverbruik · Vermogenopwekking · Vermogenfrequentie · Cos φ

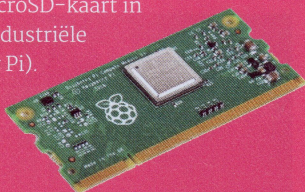
De interfaces naar de Raspberry Pi zijn elektrisch geïsoleerd om maximale veiligheid te waarborgen.

www.elektor.nl/smartpi-2

Kits & Modules

RPi Compute Module 3+ (32 Gbyte)

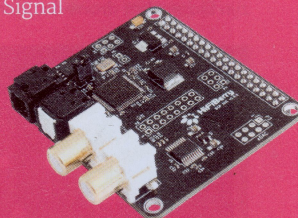
De Raspberry Pi Compute Module 3+ (32 Gbyte) levert de kracht van een Raspberry Pi 3 Model B+ (de BCM2837B0-processor met 1 Gbyte RAM) en een 32 Gbyte eMMC-flashgeheugen (het equivalent van de microSD-kaart in de niet-industriële Raspberry Pi).



www.elektor.nl/rpi-cm3-32gb

HiFiBerry DAC+ DSP

De HiFiBerry DAC+ DSP is een digitaal-analoog-omzetter met hoge resolutie voor de Raspberry Pi. Het combineert een Burr-Brown dac met digitale in- en uitgangen en een krachtige dsp (Digital Signal Processor).



www.elektor.nl/hifiberry-dac-dsp

SunFounder Raspberry Pi PiSmart Box

De PiSmart Box is een intelligent platform gebaseerd op de Raspberry Pi, dat spraakherkenning (Speech-to-Text), spraaksynthese (Text-to-Speech) en servo- en motoraansturing combineert. Geschikt om robots aan te sturen en te experimenteren.



www.elektor.nl/pismart

De beste Pi-projecten

Dit zijn de beste Pi-projecten die we de laatste tijd hebben gezien

CYBERPUNK-LAPTOP

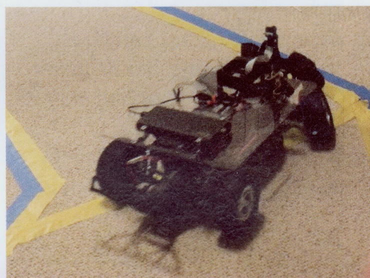
2019 lijkt wel het jaar van de cyberpunk te worden. Dat vinden we niet erg! Dit laptopontwerp dat gecamoufleerd is in een oude mahjongdoos, inclusief kleurrijke toetsen en zichtbare printplaatjes, vinden we prachtig.



▶ **magpi.cc/CCJppK**

ZELFRIJDEND AUTOOTJE

Zijn zelfrijdende auto's de toekomst? Daar gaan we ons hier niet over uitspreken, maar Pi-robots die geprogrammeerd zijn om automatisch te rijden, vinden we wel leuk. Bekijk zeker de video eens van dit kleine robotautootje dat automatisch rond het spoor op de grond rijdt.



▶ **magpi.cc/vjDimi**

PI OP HET WATER

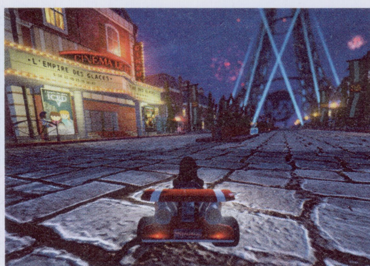
Een door een Pi aangestuurde boot met camera en afstandsbediening! Schitterend toch, hoewel we veronderstellen dat het bereik afhangt van je wifi-signaal of de lengte van een kabel en daardoor beperkt is. Maar toch, een geweldig project!



▶ **magpi.cc/ndipGc**

SUPERTUXKART OP DE PI!

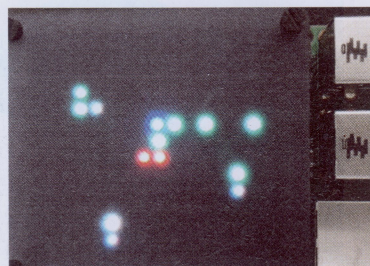
Hoewel Tux, de beroemde mascotte van Linux, niet zo populair is als Mario, heeft het sommige enthousiastelingen niet tegengehouden om hem in de plaats van Nintendo's spelfiguur te gebruiken in een opensource kloon van de Mario Kart-serie, SuperTuxKart. Je kunt er nu ook een online server op een Pi mee draaien. Leuk!



▶ **magpi.cc/JxAUZe**

BEKIJK DE VLIEGTUIGEN BOVEN JE HOOFD

Dit project met een Unicorn HAT helpt je om de hemel op elke locatie in het oog te houden. Dit voorbeeld volgt de vliegtuigen boven Heathrow: het haalt de informatie uit online bronnen en toont de levendige activiteit op een leuke manier. Dat moet spannend om te zien geweest zijn tijdens het drone-incident.



▶ **magpi.cc/pirevj**

GAME BOY ADVANCE SP

De GBA SP is een van de kleinste draagbare gameconsoles die Nintendo ooit gemaakt heeft. Toch is hij nog groot genoeg om een Raspberry Pi te kunnen herbergen. De ideale power-up dus voor je 15 jaar oude Game Boy!



▶ **magpi.cc/TzHEEf**